;login:

The USENIX Association Newsletter

Volume 9 Number 3 **July 1984**

CONTENTS

The deadline for submissions for the September issue of *;login:* is September 4

NOTICE

;login: is the official newsletter of the USENIX Association, and is sent free of charge to all members of the Association.

The USENIX Association is an organization of AT&T licensees, sub-licensees, and other persons formed for the purpose of exchanging information and ideas about UNIX† and UNIX-like operating systems and the C programming language. It is a non-profit corporation incorporated under the laws of the State of Delaware. The officers of the Association are:

President	Alan G. Nemeth	Directors	Thomas Ferrin
Vice-President	Deborah K. Scherrer		Steve C. Johnson
Secretary	Lewis A. Law		Lou Katz
Treasurer	Waldo M. Wedel		Michael D. Tilson

The editorial staff of *;login:* is:

Technical Editor	Brian Redman
Managing Editor	Tom Strong
Editorial Director	Lou Katz

Member services are provided through the Association office. Membership information can be obtained from the office:

USENIX Association
P.O. Box 7
El Cerrito, CA 94530
(415) 528-8649

{ucbvax|decvax}!usenix!office

The Association uses a VAX‡ 11/730 provided by the Digital Equipment Corporation for support of office and membership functions, preparation of *;login:*, and other Association activities. The VAX runs 4.2BSD, which was provided and installed and is maintained by Mt Xinu.

Members of the UNIX community are heartily encouraged to contribute articles and suggestions for *;login:*. Your contributions may be sent to the editors electronically at

{ucbvax|decvax}!usenix!login

or through the U.S. mail to the Association office at the address above. The USENIX Association reserves the right to edit submitted material.

;login: is produced on UNIX using *troff* and a variation of the −me macros. We appreciate receiving your contributions in *n/troff* input format, using any macro package. If you contribute hardcopy articles please leave left and right margins of 1" and a top margin of 1½" and a bottom margin of 1¼". Hardcopy output from a line printer or most dot-matrix printers is not reproducible.

---

†UNIX is a trademark of Bell Laboratories. ‡VAX is a trademark of Digital Equipment Corporation.

USENIX-Sponsored Workshops

UNIX and Distributed Systems Workshop

September 12–14, 1984

Hotel Viking
Newport, RI

USENIX is sponsoring a limited-enrollment workshop on current and future developments in distributed systems aspects of UNIX, including:

- Distributed UNIX Systems
- Applications of distributed UNIX Systems
- Discussions of recently developed UNIX facilities (such as the Berkeley IPC mechanism)
- Future directions and developments

This workshop will be structured for highly interactive sessions, with presentations that will stimulate discussions of technical issues. Because of our desire for interaction, enrollment will be limited to 120 attendees. All attendees will be expected to contribute to the workshop by short presentations or participation in the discussions. Selection of attendees was based on a brief position statement refereed by the program committee.

This announcement was posted and distributed at the Summer USENIX Conference in Salt Lake City, and via netnews in mid July. The position papers were to be 150-200 words describing what you are currently working on and any ideas of interest for the workshop. It also included *name*, *address*, *affiliation*, *phone number*, and *net address*.

The registration fee will be $175.00 and will include meals and a reception, but not hotel accommodations. Complete registration information will be mailed to the selected attendees.

Position papers were due to be submitted before the end of July to:

Rob Gurwitz
BBN Laboratories
10 Moulton Street
Cambridge, MA 02238

gurwitz@BBN-UNIX
decvax!bbncca!gurwitz

Program Committee:

Rob Gurwitz, BBN Laboratories, Chair
Steve Chapin, Hydra Computer
Mike Karels, UC Berkeley
Sam Leffler, Lucasfilm Ltd.
Kirk McKusick, UC Berkeley
Alan Nemeth, Prime Computer

UNIX and Computer Graphics Workshop

December 13—14, 1984

DoubleTree Hotel
Monterey, CA

USENIX is sponsoring a limited-enrollment workshop on current and future developments in interactive computer graphics under UNIX, including:

- Large scale graphics databases
- Real-time implementations
- UNIX as a graphics development environment
- High speed data transfer
- Future developments and directions.

The workshop will be structured to facilitate in-depth discussions of technical issues, and will have presentations in a number of formats, with ample time for questions and responses. There will be a computer graphics film and video presentation on Friday night.

In order to cover the extra expenses entailed in providing high quality visual presentations, the registration fee will be $200, which will include a reception. The hotel rate for this conference is a special $65/night for either single or double occupancy.

For further details and application information, contact the Program Chair:

Reidar J. Bornholdt
Room 7-444
Columbia University
College of Physicians & Surgeons
630 West 168 Street
New York, NY 10032

{harpo,cmcl2}!cucard!reidar or {ucbvax,decvax}!usenix!reidar

Program Committee:

Reidar Bornholdt, Columbia University, Chair
Lou Katz, Metron Computerware
Tom Duff, Bell Laboratories
Peter Langston, Lucasfilm Ltd.

Communications and Networking Workshop

October 11—12, 1984

Golden, CO

USENIX is sponsoring a limited-enrollment workshop on current and future developments in communications and networking aspects of UNIX.

The program chair is:

Doug McCallum, NBI Inc.
nbires!mccallum

Further information on this workshop will be available through the USENIX office and will be posted to net.usenix.

Future Meetings of the USENIX Association

The Winter 1985 Conference of the USENIX Association will be held January 23−25, 1985, at the Fairmont Hotel in Dallas, Texas. This conference is being held in the same city and at the same time as the /usr/group sponsored trade show, UniForum. Arrangements for cross-registration for those wishing to attend both events are being discussed with /usr/group.

The Summer 1985 Conference will be held June 11−14, 1985, in Portland, Oregon.

Future Meetings of Other UNIX Users Groups

Australian UNIX-Systems Users' Group

The 1984 winter meeting of the AUUG will be held in the Department of Computer Science at the University of Melbourne on August 27 and 28, 1984. An exhibition of UNIX-related computer hardware and software will be held in conjunction with the meeting. The conference dinner will be held Monday; reservations for it must be made with advance registration. A limited number of rooms are available in the University Colleges. The meeting cost will be $30 for advance registrants, $50 on site. For more information, contact

Robert Elz
Dept. of Computer Science
University of Melbourne
Melbourne, Vic., Australia

Phone: (03) 341 5225, International +61 3 341 5225

or

Prue Downie
(same address)
Phone: (03) 341 5232

European UNIX Users Group

The EUUG and /usr/group are sponsoring a conference and exhibition at St. Catherine's and Pembroke Colleges, Cambridge University in the United Kingdom on September 19−21, 1984. The invited speakers are: John Lions, Steve Bourne, Pier Dick-Lauder, Tom Killian, Mike Karels and Sam Leffler.

The entrance fees are: technical sessions only − UK*L* 90 (students 70); technical and industrial sessions − UK*L* 120 (students 100); plus 15% VAT. USENIX members pay the same rate as EUUG members. Nonmembers fees are UK*L* 40 additional.

Accommodations are available at the College Residents. Registration and requests for accommodations should be received by the EUUG Secretary before September 3. For information contact:

Mrs. Helen Gibbons
Owls Hall
Buntingford, Herts. SG9 9PL United Kingdom

Phone: 0763 73039

System V Performance Enhancements

Ken Goodwin
Senior Systems Programmer
N.J. State Medical Underwriters, Inc.
2 Princess Road
Lawrenceville, NJ 08648

Described below are a series of modifications made to the UNIX System V kernel and several commands. Changes to the kernel have resulted in a 12% decrease in system overhead, while command changes have increased usability and security. All changes were made on a DEC PDP-11/70 minicomputer, but are applicable to any UNIX system. Percentages and times were obtained using the *clock* subroutine under single user conditions.

User Level DMA I/O

A facility that allows user programs to specify Direct Memory Transfers for disk operations involving **regular** files has been implemented. It is enabled through a bit flag in the *open* or *fcntl* system calls. DMA operations may occur for any read or write operation that begins on physical disk block boundaries and whose transfer size is at least the same size as a physical disk block. However, no restriction is placed on the user program to insure that its transfers meet these criteria. As the I/O request is broken down into transfers to specific blocks by *bmap*, a decision is made as to whether a DMA transfer can take place from/to the disk block. If a DMA transfer can not take place, either because the transfer does not start at a zero block offset or the count of bytes remaining to be transferred is less than the physical size of the block, then the transfer is done through the Buffer Cache as before. If a DMA transfer can take place, then a call to the *dma* subroutine is made. The *dma* subroutine checks the request for validity, initiates the I/O operation asynchronously through the *phybio* subroutine, a modified version of the standard *physio* subroutine, and then updates the count, offset, and base fields in the Per Process Data Area (_u). While this transfer takes place, *dma* checks to see if a Read-Ahead transfer can also be done. This is determined by the presence of a read-ahead block number in *dma*'s parameter list and if the updated count field is still greater than or equal to the physical size of a disk block on the filesystem. If read-ahead can be done, another asynchronous I/O operation is queued through the *phybio* subroutine and the count, base, and offset fields are again updated. *dma* then waits for all the I/O requests that it has initiated to complete. On errors, the count, offset, and base fields are backed up to the point they were at before the error occurred and u_error is set. In this way, up to two disk transfers can be completed per invocation of the *dma* subroutine. This has the effect of minimizing the number of *dma* calls performed and halves the number of loops performed within the *readi* and *writei* subroutines. Changes required to implement DMA I/O were minor. They consist of:

1. A modified version of *physio* (*phybio*) which permits asynchronous DMA queueing of I/O requests between the Disk and user I-Space, D-Space, Kernel I-Space, Kernel D-Space, Supervisor I-Space, or Supervisor D-Space.
2. A *phywait* subroutine implements the *iowait* and *error* functions from the *physio* subroutine.
3. A *dma* subroutine which handles the queueing of the requests, updating the relevant fields, and dealing with error conditions.
4. Changes to the *readi* and *writei* subroutines which allow them to determine if DMA I/O is desired and permitted.

5. A single line change to *bmap* to turn off Read-Ahead calculations if DMA I/O has been selected and the next block in the file is the last block and is partially filled.

This facility has been added to the *cp*, *dd*, and *cpio* commands. For any filesystem using 512 or 1024 byte disk blocks, the performance increase for programs using this facility is approximately twenty-five percent. It also has the added advantage that corruption of the buffer cache by these programs no longer occurs and cache hit ratios thereby increase for other programs. This facility is not for use by every program. Candidates for DMA I/O include programs such as *cp* which never reference a disk block more than once. A possible modification to the stdio library to allow the selection of DMA I/O operations is anticipated. Since stdio already does buffering, there is little need in most cases for additional buffering by the kernel. Higher performance increases are possible with larger physical block sizes. A port of the algorithm to a Berkeley 4.2BSD system is planned.

Faster *exec* System Call

The *exec* system call now uses the new DMA I/O facility to directly load programs into their process space. A great deal of memory-to-memory overhead is thereby avoided. The performance increase for *exec* is greater than thirty percent. Since *exec* does not have the system call overhead of the standard *read* system call, it gains a greater performance improvement than a user level DMA operation. No actual timing comparisons were made to calculate the performance increase for *exec*. For user level DMA I/O requests, each doubling of the read count resulted in a one percent performance improvement. This has been extrapolated to yield the thirty percent figure based on the analogy that an *exec* call translates to a single *read* call of process-size bytes. The performance increase may therefore be much greater.

Improved Read-Ahead

A change to the standard method of calculating read-ahead for files has been implemented. On standard UNIX systems, the last logical block referenced in a file is stored in the incore inode structure as i_lastr. Although this is quite adequate for One-Program One-File scenarios, it has the tendency to turn off read-ahead all together when more than one process is referencing the same file. The *bmap* subroutine determines that read-ahead should be initiated if i_lastr+1 is equal to the current block that the process wants (i.e., the file is apparently being read sequentially). If two processes are reading the same file sequentially, but are several blocks apart in the file, then read-ahead is turned off for the process that is further along in the file. This is due to i_lastr being flip-flopped between the last block read by each process, such that i_lastr+1 never equals the block number desired by either process. The trailing process gains only partial read-ahead. The amount of this read-ahead depends on how fast other processes gobble up buffers containing the blocks that this process has yet to reference and how much further along in the file the leading process is. If the leading process is more than NBUF blocks ahead of the trailing process, then read-ahead will be totally turned off for both processes.

This standard algorithm has been modified to allow read-ahead calculations to be computed on a per-process basis. The i_lastr disk address has been replaced by a u_lbr disk address and an array of disk addresses u_lastr[NOFILE] in the per-process data area (_u). The *rdwr* subroutine has been modified to copy the contents of u_lastr[FD] to u_lbr before calling *readi* or *writei* and to copy u_lbr back to u_lastr[FD] when *readi* or *writei* returns. The *dup* and *fcntl* system calls now copy u_lastr[OLDFD] to u_lastr[NEWFD] and *open* sets u_lastr[FD] to zero. The *bmap* subroutine now uses u_lbr instead of i_lastr. This modification has the added advantage of freeing up much needed data space since the Per Process Data Area is not a permanent physical part of kernel data space while i_lastr was. The amount of data space released is NINODE*sizeof(daddr_t). The *exec* system call sets u_lbr to zero. This is done to allow read-ahead to proceed if DMA *exec*'s are not used. There is no performance change for the One-Process One-File scenario, but this change increases the read-ahead performance of the Many-Process One-File scenarios that are an integral part of large scale database systems.

Process ID Map

The standard Process ID (PID) allocation method in UNIX is to compare the desired new process id against the process ids already allocated in the Process Table. At best, this involves doing (v_eproc−&proc[0]) comparisons during the first MAXPID processes. On any system which manages to age past MAXPID processes, the number of comparisons performed greatly increases. This is due to PID collisions that begin to occur which make the code branch out of the process table scanning loop in order to select an alternate PID. This new PID may also collide.

This allocation method has been replaced by a Process ID map strategy. The process table scanning is now reduced to a single linear search to locate an empty slot and to count up the number of active processes associated with the Process group. This eliminates the PID retry code above the process table scanner and an IF−GOTO statement within the process table scanner. This IF−GOTO statement consumes about 2.7 microseconds on a PDP-11/70.

Originally, there were two process ID maps. One was used to hold process IDs available for use and the other was used to hold process IDs that had already been used. Two pointers were used to reference one as a New PID map and the other as an Old PID map. These pointers were initialized during system generation.

```
struct map pid1[PMAPSIZ] = {mapdata(PMAPSIZ)};
struct map pid2[PMAPSIZ] = {mapdata(PMAPSIZ)};
struct map *newpid = &pidmap1;
struct map *oldpid = &pidmap2;
```

The newpid map was initialized during system boot by an *mfree*(newpid, MAXPID, 1) call in *main*. Since the size of items allocated and freed in the process ID maps is always of size one and in order to minimize overhead, special versions of the *malloc* and *mfree* subroutines were created. They are called *pid_alloc*(map) and *pid_free*(map, pid). The *exit* system call freed used PID's into the oldpid map when a process died, *pid_free*(oldpid, p_pid). This insured that new process IDs remained unique for as long a time as possible. The *newproc* subroutine used *pid_alloc*(newpid) to allocate a process ID when creating the new process. When the newpid map was exhausted, a switch of the newpid and oldpid pointers was made. This moved the contents of the oldpid map into the newpid map and reinitialized the oldpid map.

```
struct map *swmap;

swmap = newpid;
newpid = oldpid;
oldpid = swmap;
```

Another *pid_alloc*(newpid) was then done and should succeed. A panic situation was generated if this *pid_alloc* failed. However, the only way this could occur is if something overwrote the maps or if the *pid_free* in *exit* could not free a significant number of old PIDs into the oldpid map because PMAPSIZ was too small. PMAPSIZ had to be roughly 30% greater than NPROC to allow for long term fragmentation.

For oldpid and newpid maps which were fragmented and contained around seventy entries, the break-even point for this algorithm was 66 active processes in the process table. If standard *malloc* and *mfree* subroutines are used to manage the PID maps, then the break-even point rose to 71 active processes. This break-even point was only for the first MAXPID processes. It declined once MAXPID was exceeded since collisions would begin to occur at this point under the old algorithm. This algorithm was obviously not for small UNIX systems.

Note: two microseconds were trimmed off the average *malloc* subroutine call by changing the *malloc* code to put the size parameter in a register.

The times below are in microseconds as computed from *clock* subroutine calls. The tests were run single user and are adjusted for extraneous times caused by instructions used to run the timing

tests. "Frag." is short for fragmented and indicates that test was run on a fragmented map of about 70 members. If not specified, then the test was run on an unfragmented map. "Alloc" is for a *malloc* or *pid_alloc* call and "Free" is for *mfree* or *pid_free* calls. "Breakeven" is the number of active processes that must be present for the map strategy to break-even over the old algorithm during a single scan of the process table. Breakeven = Total Frag. / 2.7μs. "Aver." indicates the average overhead of making the subroutine call on an unfragmented map. The column headers are: "Old" – using standard *malloc/mfree* subroutines, "New" – using improved *malloc* subroutine, "Pid" – using *pid_alloc/pid_free*.

Test Times

Test	Old	New	Pid
Aver. Alloc	34.8	32.8	29.8
Aver. Free	47.75	47.75	42.19
Alloc Frag.	35.54	34.43	30.53
Free Frag.	153.84	153.84	146.6
Total Frag.	189.38	188.27	177.13
Breakeven	70.14	69.73	65.6

Note that the overhead of a free operation on a fragmented map is seventy eight percent of the total time in this strategy. This was deemed unacceptable and the problem was subjected to intense analysis. The above PIDMAP algorithm was modified as follows. The *pid_free*(map, pid) was removed from *exit* since this represented seventy eight percent of the overhead. The data structures were changed to:

```
struct map pidmap[MAPSIZ] = {mapdata(PMAPSIZ)};
int usedpids[NPROC+2];
```

PMAPSIZ should now be NPROC+4. The *pidmap* is still initialized in *main* by a *mfree*(pidmap, MAXPID, 1) call. *newproc* calls *pid_alloc* to allocate a process id (p->p_pid = pid_alloc();). *pid_alloc* now does all the dirty work. If the pidmap has not yet been exhausted, then it returns the next available PID. Otherwise, it regenerates the pidmap from the contents of the process table. The process table is scanned and every p_pid that is greater than or equal to STARTPID is placed into a slot in the usedpids array. STARTPID is a define for the first PID which is considered in regenerating the pidmap. It is set to two on my system since PID number 1 is always in use by *init*. It represents the lower bound for the range of recurrently usable PIDs. MAXPID is the upper bound for this range. The usedpids array is then sorted via a shell sort and the sorted contents used to generate the correct entries in the pidmap. Since this operation is only performed once in (MAXPID−NPROC) process generations, great speed was not deemed necessary. Once the pidmap has been regenerated, then *pid_alloc* returns the next available PID as before. This modification resulted in an eighty three percent performance increase over the previous pidmap strategy.

The following test results were obtained by simulating the algorithm under three types of loads. "WC" is a worst case scenario: NRPROC was 400, the number of filled slots in the process table was 350, and ve_proc was set to the address of proc[NPROC] so that every process table slot was examined. "BC" is a best case scenario: NPROC was set to 200, the number of slots filled was 150, and ve_proc was set to the address of proc[150]. "NC" is a normal case scenario which simulates the tests run under the original PIDMAP strategy by forcing the regenerated pidmap to have seventy entries: NPROC was set to 100, number of processes was 70, and ve_proc was set to the address of proc[70]. For all tests, the process table was initialized with non-sequential decreasing value PIDs. This was done so that the maximum size pidmap would result and the sort would be forced to totally invert the contents of the usedpid array. Because of this, the results below should be higher than what would be obtained in actual use. Under live conditions there should be many sequential PIDs in the process table.

"Regen Time" is the time necessary to regenerate the pidmap once it has been exhausted. "Alloc Time" is the average time necessary to fetch a pid from the pidmap while it still has entries in

it. "RT-pid" is the average time to regenerate the pidmap per pid gained (Regen Time / (MAXPID−nproc)). "Breakeven" is the number of process table entries that have to be occupied in order for the algorithm to breakeven with the original linear search. Breakeven = (RT−pid+alloc. time) / 2.7μs.

Test Times

Test	WC	BC	NC
Regen Time	33294.65	16638.65	16640.65
Alloc Time	37.35	27.35	25.35
RT-pid	1.12	.56	.56
Breakeven	14.25	10.34	9.6

Note that the average breakeven point is ten processes in the process table. Small UNIX systems can now take advantage of a PIDMAP algorithm. Also, this new strategy consumes less data space than the original PIDMAP strategy. In case the high Regen times on the pidmap are causing you concern, please note the following. Suppose we assume that we maintain ninety processes in the process table during the first MAXPID processes and the breakeven point is ten. Also, because PIDs have yet to wrap around, we will experience no PID collisions. This means that each *newproc* call will be saving 80 times 2.7 microseconds of CPU time for a total of 216 microseconds (90 processes minus 10 required to breakeven). With MAXPID set to 30000, which is standard for System V, this means we will use up 30,000 PIDs before exhausting the map and forcing a pidmap regeneration. This means that the total CPU time saved before regeneration is 30,000 times 216 microseconds, or 6,480,000 microseconds. After the first regeneration we will be ahead of the original algorithm by over 6,463,359.35 microseconds.

One final note, this is not the optimum PID allocation algorithm. The optimum algorithm could not be implemented on my system as it would require 3,752 bytes of data space. Such data space is hard to come by in a PDP-11/70 System V kernel.

Optimum PID Allocation Strategy

The above algorithm can be vastly improved by replacing the memory map structure (struct map pidmap) with a **bitmap** where each bit represents the current status of a PID. An **off** bit state would indicate that the PID was either in use or had been recently discarded by *exit*. A **on** bit state would indicate that the corresponding PID was available. The bitmap is represented by an array of integers.

```
# define BPINT      16        /* number of bits in an integer */
# define BITON      0177777   /* All bits in an int on (can be -1) */
# define PMAPSIZ    ((MAXPID + (BPINT − 1)) / BPINT)

int       pidmap[PMAPSIZ];
int       *pidoff;
```

pidoff is an integer pointer used to indicate the first non-zero slot in the pidmap array so that a linear search will not be necessary. Initialization and regeneration are now easy. First one turns on all bits in every integer, then one clears the bits that correspond to PIDs already in use (including PID zero), and finally pidoff is set to the first non-zero entry of pidmap. To select a PID, the contents of the word pointed to by pidoff are scanned for an **on** bit. This bit is cleared and the new pid is

((pidoff − pidmap) * BPINT) + (bit offset in *pidoff of cleared bit)

If *pidoff is now zero, then pidoff is incremented to the next non-zero entry. When pidoff exceeds the upper bound of the pidmap array, then the map is regenerated as described above. Most of the above can be implemented as macro calls. Since the entire algorithm can be implemented as in-line code in *newproc* the subroutine overhead is eliminated. Also, no sort is required. This should therefore be the fastest algorithm.

Dialup Line Security

getty

The *getty* command has been modified to allow a parenthesized ENVIRONMENT list to be included as part of it's command line. This was done in order to ease implementation of the dialup security feature in *login*. The TERM environment variable is set for each *getty* line in the *inittab* file for the specific terminal connected to the line. For dialup lines, the TERM variable is set to dialup.

```
/etc/getty ... '(' TERM=vt100 TERMDEV=/dev/tty00 ')' ...
/etc/getty ... '(' TERM=dialup TERMDEV=/dev/tty01 ')' ...
```

For environments with various terminal types, this allows naive users to float from terminal to terminal without worrying about the type of terminal they are logging onto.

login

The *login* command now passes the environment from *getty* down to the shell. It also examines the TERM variable to see if it is a dialup. If TERM is a dialup, then a dialup line security system is invoked. This change replaces an undocumented security feature that was present in the original System V *login*. An */etc/dialups* file which has the same format as the */etc/group* file has been created. It is divided into four colon-separated fields. The first field is the device name of a phone line (*/dev/tty*??). The second field is an encrypted password field, the third field is a privilege level field, and the last field is a comma separated list of login names for users permitted to use this dialup line. If TERM is dialup, then TERMDEV is compared against each line in this file. If no match is found, then no security check is done. Otherwise, the name (*/dev/tty*??) of this line is displayed to help in answering the password or in determining why permission to login was denied. If the password is present, then the user is prompted for the password to this line. If the privilege field is present, then the user's privilege level in the */etc/passwd* file (currently group number) must be less than or equal to the privilege number in the dialups file for this line. If the list of users (field four) exists, then the user's login name must be on that list. These features may be used in any combination. If the user fails any of the security checks, then a permission denied notice is printed and the system hangs up the phone line. This can make it very expensive for someone to try to break into the system. If the user succeeds in passing all the dialin security checks, then *login* proceeds with the user name and password checks. At maximum protection level, a person must know the name and password for an account on the system, must know the password for the dialup line that the person is authorized to use, and must know it's telephone number. Combined with IMM modem security devices, this system provides a more formidable obstacle to intruders than standard UNIX systems. It also allows administrators to assign dialups to particular users or groups to prevent one group from hogging the phone lines.

A modified version of the *passwd* command handles passwords for both the *group* and *dialups* files. Modified versions of the *group* file subroutines provide access to the *dialups* file.

A *dialname* shell command file is called by the shell during the login procedure if TERM is dialup. This shell allows naive users to set their terminal type (TERM) when dialing into the system in a user-friendly manner. It prompts for the *termcap* terminal type and compares it against a known list of terminals attached to the system. If the terminal specified is not on the list, the user is allowed a second chance to change the entry. This second chance is not checked and the entry is assumed to be correct. This allows users using terminals not known to the system to log in properly. If done frequently, then the System Administrator should be requested to add the terminal type to the known terminal list. A RETURN entered to the prompt generates a two column list of known terminals and their *termcap* equivalents.

On the Design of the UNIX Operating System

Peter Collinson

Computing Laboratory
University of Kent,
Canterbury, Kent, UK
ukc!pc

ABSTRACT

An extremely inaccurate view of the history of the design of the UNIX operating system is presented.

Why is UNIX Successful?

The computer world seems to have gone "UNIX mad", and it is hard to understand why. One good reason is the portability of the system but there must be more to it than that. Most people who use the UNIX system seem to like it even though it is full of idiosyncrasies, is terse to the point of unhelpfulness and consists of a very large number of totally forgettable commands. I think that the success of the system is summed up by the following paragraph.

> The UNIX system is successful because the minimum number of keystrokes achieve the maximum effort. In addition, the system says very little to explain errors and relies on the intelligence of the user to deduce reasons for failure.

The statement describes UNIX V6, which we all know is the parent of the UNIX systems running today. History tells us that the guys who designed it did their own typing into the machine. It seems to me that because of this, the main reason that UNIX enjoys/suffers from terse input and output is not through any intellectual design decisions made at some early stage but because the UNIX designers were just bad typists working on slow peripherals.

Let us examine the evidence.

Ancient History

First, there are all those incomprehensible and forgettable two letter commands: *ls*, *cp*, *mv*, *pr*, etc, etc. What delight it was to type three letter commands: *cat*, *cdb*, and *dsw*. It is interesting to note that if you were to use the text formatters, it was assumed that you could type. So, to use *nroff* you had to make five painful keystrokes, admittedly only four different letters were used. Of course, all the commands were in lower case because the designers couldn't find the shift key on the keyboard, never mind the shift-lock.

Secondly, look at the main means of the input of text, another two letter command, *ed*. Every command in the editor is a single letter, which are just about mnemonic. Every error message is a single letter, '?', which is just about meaningless‡. However, only a very few commands are needed to do basic editing operations. If the user wants to do something complicated, then something complicated must be typed in. This complicated thing is often very short and incomprehensible to anyone other than the author. Is there anything more minimal than regular expressions in *ed*? Learning to use *ed* is a continuous voyage of discovery, users find that less and less typing is required as time goes on. What joy it is to finally find out how \(is used. *ed* is written in C and is available for hacking, and the alterations made to it by many hackers in many places were thought to be required to make the editor more powerful. That is, you could do more in less keystrokes. So, *ed* is aimed at bad typists, and it developed to support bad typists.

‡OK folks, I didn't forget about the immortal TMP error message and the fact that '??' means something special.

C is another example of design for bad typists; *integer*? – no, *int*, *structure*? – no, *struct*, *begin/end*? – no, '{ }'; ':='? – no '=', after all it's easy to type '==='; long variable names? – yes, but only 8 characters were significant, so it wasn't worth typing any more; and the list could go on and on. C is so terse that it is enigmatic, and was obviously designed for coding by bad typists who are actually typing the code themselves. Most other languages seem to be designed for people who are paid by the number of inches of code they write on paper for other people to input into the machine.

Apart from the famous patented setuid bit, the notion of the use and exploitation of software tools is the main feature of the UNIX system which has become important and perhaps even academically respectable. But, did those guys really sit down over a cup of coffee one day and say "hey, I've had this really neat idea, let's code a bunch of simple programs onto this experimental file system, provide some way of joining the output of one program to the input of another – and yes, a good name would be UNIX." Heck no, one of them said to the other "I'm really up to here with this typing, can't we use the output of that program into the input of this, it'll save me lots of time, which I can spend in playing some really neat games." So, software tools were born.

So, the evidence is very strong that bad typing played a large role in the design of the UNIX system. With its terse input and terse output, UNIX V6 was a joy to use – I am a bad typist too.

More Recent History

A number of improvements were seen in UNIX V7. Notably, better typists were employed to design and implement things. The Bourne shell, *sed* and *awk* are examples of this. Those original guys had got a bit better typists too, there are a few more comments in the kernel and some longer names. C had altered and generally requires more typing. All the new things have long names: *typedef*, *unsigned* variables, and the type checking of function returns meant the input of function specifications at the start of the program.

UNIX was ported to a number of machines using the very wordy portable C compiler. However, ports of the system to machines with long names were not successful and the enduring port was to the VAX architecture, because VAX is short enough for everyone to type. And then came the Computer Systems Research Group of the University of California, Berkeley.

It is very fashionable to be rude about UCB systems these days. This rudeness all stems from typing envy, because those people at UCB can really type. It seems that applicants for posts at CSRG, UCB have to pass a typing test to get in; the test consists of typing the full title of the institution without using the delete key.

These really great typists contributed mightily to the development of UNIX. For example, they virtually invented the useful comment in the kernel code. It is suspected that the people who understood the comment *You are not expected to understand this* don't understand why comments are useful. However, UCB hackers tend to show off their skill somewhat and embarrass others by producing really big programs. Most of the rudeness about UCB system stems from the scorn poured onto these big programs, people seem to forget that other UNIX influences have generated really big programs. For instance, you can't get the original *f77* V7 compiler into a PDP-11 unless it has separate I/D space.

AT&T headed towards re-organisation and the oddly named UNIX System III made its appearance. It was odd, too, because it is based on the PWB UNIX system which not many people had used. It didn't feel like UNIX because lots of things had moved and had their names changed. With System V, AT&T tell us that this is **THE** UNIX operating system and hopes it will rule the world. On initial inspection, System V is well typed and has long manuals. Error messages are now wordy and it is almost possible to say that you don't need the source to find out why something happens the way it does. In truth, the current situation is that the UNIX systems which the outside world can obtain are in the hands of people who employ others to type – or so it seems. Typists have moved into the UNIX world, and those of us who are still bad typists complain that UNIX V6 was the last real UNIX system.

Meanwhile, back in Bell Labs, those guys have conquered the typing problem by designing high quality terminals where all you need to do is point. But that's another story.....

European UNIX System User Group Meeting Report

Faculty of Science
University of Nijmegen
16–18 April 1984

Peter Collinson
Secretary

Introduction

This was a very good conference, and if you missed it, then you missed a large number of very good talks, the biggest exhibition at a EUUG conference to date and, of course, some Dutch beer.

This report is being done from my incomprehensible notes and mostly from the abstracts which were submitted before the conference began†. I must confess to missing some of the sessions; still, that is always going to happen. I also feel that this report is not up to the usual standard because I am doing it too long after the event and this makes it difficult to precis the talks. So, where I am in doubt I have kept quiet. If have got any names wrong, then I am sorry.

However, the intention is to publish a full conference proceedings in the near future. Meanwhile, this can serve as a summary of what happened.

Day 1 – 16th April 1984

Emrys Jones officially opened the conference and chaired the first session.

Item 1: 10.00am **Michael J. Kelly, AT&T/Teletype Corp**

An Intelligent Windowing Graphics Terminal for the UNIX system

Abstract

An important feature of the UNIX System is per-user multiprogramming; that is, each user may control several concurrently executing processes. However, this feature breaks down at the user interface. UNIX systems rely on "dumb" or semi-smart terminals as the primary user interface, and these are not able to maintain several concurrent interfaces. The solution found in BSD, job control, still does not adequately solve the problem of maintaining display contexts for concurrently executing processes.

This talk was about AT&T 5620, the Teletype Corp's version of the Blit terminal. It is a workstation with a high resolution green screen, a mouse and has a reasonably powerful machine to drive it. The machine does not run UNIX because the device is a terminal and not a working CPU in its own right. The processor is a WE 320001 CPU with 64K bytes of ROM, 256K bytes of RAM expandable to 1 Megabyte. The terminal also has an RS232 interface.

The main power of the terminal derives from an interface to UNIX System V, which allows several windows to be placed on the screen. Each window has its own control software running in the terminal and also a user level protocol handler running in the host.

Q. Is the protocol specified and available?

A. Not yet

Q. Are there any cross development tools for the 32000 CPU?

A. Yes.

Q. What is the availability in Europe?

---

†Thanks to Japp Akkerhuis for finding them in a machine readable form and sending them to me.

A. Olivetti will distribute it.

Q. Will the software run on any UNIX system other than System V?

A. No comment.

Item 2: 10.30am **P. Freund, Hewlett Packard**

A Layered Implementation of the UNIX Kernel on the HP9000 Series 500

Abstract

An implementation of the UNIX operating system kernel has been layered on top of an existing operating system kernel for the HP9000 series 500 computer. The mapping of UNIX functional requirements onto the capabilities of the underlying OS are presented in this paper, including the changes and extensions necessary to support UNIX semantic and performance requirements. The paper covers in retrospect the advantages and disadvantages of a layered approach.

This talk discussed HP's approach to the implementation of UNIX System III (with those familiar Berkeley enhancements — i.e. **vi**). The hardware is based on a single chip 32-bit CPU with a stack based architecture. A configuration can consist of several multiple CPU's. HP wanted to support operating systems other than UNIX, and have developed an internal kernel called SUN (no relation to those other folks, folks) onto which UNIX has been layered.

The SUN kernel has memory management, process management, a file system supporting multiple directory formats, device drivers, some I/O primitives, a real-time clock and interprocess message passing. It does not have a human interface because is it designed to support operating systems and not humans. Also, there is no means to load programs.

The talk then described the ins and outs of the implementation which I hope will be reproduced elsewhere. I was left with the feeling that the system worked, but perhaps someone out there in UNIX-land might like to give a slightly less biased report.

☛ Coffee (and a peek at the exhibition) ☚

Session 2 was chaired by Adrian Freed, Ircam, France.

Item 3: 11.31am **Andy Tucci, Amdahl Corp.**

Future Directions for UNIX at Amdahl

Abstract

This talk will cover future plans for UNIX on Amdahl.

The talk started by summarising the history of UNIX running on Amdahl machines. The system is called UTS and runs in a virtual machine under the VM operating system, the latest release is UTS 2.2 and UTS 2.3 will be released shortly. The current release has "good V7 compatibility." The idea of marketing UNIX is to take advantage of an internal product which had been developed for in-house engineering use, to increase their reputation as a software vendor rather than their being known as simply a hardware supplier, and to make inroads into the academic community. Future plans are to: continue leadership in the large system UNIX market and to maintain compatibility with the Bell Labs product at current release levels. I.e. they are working hard on System 5, which is running in-house and at some installations of early customers.

Q. Pricing?

A. An AT&T license is required, plus $1500 per month.

Q. European support?

A. Yes.

Item 4: 11.53am **Larry L. Crume, AT&T Bell Labs**

UNIX* System V Release 2.0 and Future Directions

Abstract

Enhancements to UNIX System V Release 1.0 will be reviewed. These enhancements which are to be released in April 1984 include feature updates and improvements in the following areas: C Compilation System; Job Control/Virtual Terminals; Shell; Commands; Cron Facility; Curses/Terminflo Package; Standard Disk and Tape Names; Accounting Package; Performance; and New Documentation.

Future Directions: Future enhancements to the UNIX System will focus on the user interface. The paging system under development at AT&T Bell Laboratories shows one dimension of the user interface for systems programmers. The architecture must be general enough to support both paging and swapping kernels and many different memory management units.

The work on command syntax and error message handling provides for a consistent user interface and to easily determine and recover for errors.

Work on unbundling and repackaging the UNIX System will provide for a consistent user view of the UNIX System.

To fill that in a little. This year's system from AT&T is called System V, release 2. There are several new applications programs which come with the system and some enhancements to the operating system.

The biggest development objective is to maintain upwards compatibility while improving performance. This does not necessarily mean that the increased speed for increased size trade-offs will be done, but AT&T are looking at some of the UCB enhancements.

Job control has been introduced. This has been done in a totally different way from the UCB implementation because it was not desirable to introduce the necessary new signals. AT&T are committed to support and not extend the current set of system calls. Job control is done by having a number of layers which define virtual terminals. On login, the user talks to a control layer and has the ability to start a number of shells in other layers. Input and output from these virtual terminals can be controlled independently.

The new C compiler will have long variable names. Also, there will be a C cross compiler for the M68000.

Larry then talked about the current ideas of unbundling UNIX. The pieces will consist of a basic set of utilities plus the kernel and yes, your favourite command is bound to be missing. There will then be several add-on pieces, such as the C language tools, the various workbenches, and other utility sets.

On the list of things to be looked at in future releases include: record and file locking, this will initially be the /usr/group standard; bad block handling; and file system integrity, such as protection from unexpected halts, ordered writes to discs, timely flushing of buffers and improved detection of corrupt file systems. A big thing on the list is a paging kernel in order to provide a large address space. Here, the main aim is to not affect users who don't wish to have paging in their machine. To this end, an architecture for memory management has been developed, with the idea that the paging should be easily configurable in or out. This is not on the current release, because it wasn't good enough.

This was an interesting talk, really, packed with a lot of stuff which I feel there is no room to put here. AT&T seem to be very responsible with the developments which they propose, it is perhaps possible that they are trying to generate too many new systems and are not leaving things to settle enough. My other main criticism is that most of the things coming out are fairly mundane and ordinary, the interesting leading edge of technology stuff is staying under wraps until it becomes safe and boring. I suspect that as an academic user, I would like to have Edition 8.

☛ Lunch (and a bottle of beer) ☚

This session was chaired by Bjorn Eriksen.

Item 5: 2.05pm **Bill Murphy, AT&T International**

UNIX Licensing

Abstract

What you can and cannot do for your $43,000, $68,000 etc.

Well, all Bill did was show his face and get off. The idea was that people could tackle him outside the main hall. Which I suppose they did.

Item 6: 2.08pm **Robert Ragan-Kelley, Pyramid Technology Corp.**

OSx: Towards a Single UNIX System for Super-minis

Abstract

OSx is a dual-port of 4.2BSD and System 5 onto the pyramid 90x computer, a high-end super mini. OSx is designed to be fully compatible with both 4.2 and System 5 in a fashion that neither suffers performance penalities from the coexistence of the other. This paper discusses some of the details of this design, both internal to the kernel and at the user interface level, along with some of the problems we faced in its implementation.

The idea is to implement "universes", one called **btl** and the other **ucb**. These two names are used as commands to switch between the two universes. It is also possible to have a command line containing commands from one universe in the other, by prepending the alien command by the appropriate universe name.

The implementation started from 4.2BSD and emulates System V. The main reasons for starting from 4.2BSD are the demand paging, the fast file system, flexible file name lengths and the larger block size. Also, the 4.2BSD networking would be difficult to implement on a System V.

The main problems are: the differences in the directory structure, System V FIFO's (named pipes), signal handling, System V IPC and worst, the differences in terminal handling and **ioctl**'s.

Item 7: 2.40pm **Eric Allman, Britton-Lee, Inc.**

The Special Advantages and Difficulties with Databases in UNIX (1)

Abstract

Many applications maintain some long term state in the form of a database. In many cases ad hoc algorithms are sufficient (e.g., sequential scans of the password file are adequate on most systems), but often more sophisticated algorithms must be considered (e.g. mailboxes must be locked while mail is being delivered; the dictionary is too large to be practically searched sequentially).

Although ad hoc approaches are acceptable for small applications, larger applications often find it convenient to utilize a full-blown database system. Such systems may include such features as efficient access methods, logical independence from data structure, aggregation, protection, integrity constraints, multi-file capability, concurrency control, crash resilience, audit trails, and transaction control.

The structure of a database system incorporating most of these features is examined. Interfaces, data models, cost/performance tradeoffs, and the special advantages and difficulties UNIX offers to database systems are discussed.

This, the first of two talks, gave a general introduction to database terminology and operations. It seems to me that it would be silly to attempt to summarise his talk here, we'll wait for the paper which will no doubt be considerably more comprehensible than anything I can write.

☛ Coffee ☚

Item 8: 3.42pm **Eric Allman, Britton-Lee, Inc.**

The Special Advantages and Difficulties with Databases in UNIX (2)

The second part of the talk was concerned with the facilities which are provided by data base systems and the trade-offs inherent in such systems.

Item 9: 4.01pm **P.B. Pynsent, University of East Anglia**

The Norwich Renal Unit Programme

Abstract

In Europe 120 people per million of the population suffer from chronic renal disease and of those 80% depend on an artificial kidney machine for survival. We have developed a UNIX based computer system which not only provides access to a patient database but also controls kidney machines during the haemodialysis of patients.

The objective of the Norwich Renal Unit project is to improve patient care using computer technology. First we have provided facilities for computer controlled kidney machines to optimise dialysis therapy to the individual patient and secondly we have provided an easy to use patient database to aid the physician in his assessment of patients. The UNIX operating system has proven an ideal environment satisfying both the multitasking and data processing requirements of our project.

I really liked this talk. It is so rare at UNIX gatherings to see people who are doing something real with computers. I think that I mean that the majority of talks at EUUG conferences are really "Computer Science" and I would like to see more applications oriented presentations.

Item 10: 4.31pm **Andrew Hume, AT&T Bell Labs Computer Research**

Eric: An Experimental Information Manipulation System

Abstract

Eric is a testbed for a model of how the user interacts with a computer system. The major components are the filing system, multi-tasking, the use of forms as the only means of data input and a user interface dependent on a bit mapped graphics display.

The abstract does not do justice to the talk, but my notes are even worse because I spent most of the time concentrating on what was being said.

☛ End of Day 1 (and onto the Hotel Erica for free drinks) ☚

Day 2 — 17th April

Well, I managed to make it out of bed to chair the first session of the day.

Item 11: 9.39am **Kirk McKusick, University of California, Berkeley**

The Dynamic Profiling System

Kirk's talk centred on the new profiler **gprof** which comes with 4.2BSD, and described how he had been using it to improve kernel performance. The original UNIX profiler presents a flat profile of program performance with the emphasis of the amount of time spent in a particular routine. **Gprof** does better than this by tracing the path through the code and giving statistics for: how often routines are called; from where; and in turn, which routines were called by the routine under consideration. (Having used it to analyse program performance, it's really good).

Kirk then described a scientific investigation into UNIX kernel performance. He found that **namei**, the main directory search routine in the kernel took one quarter of the system time in 4.2BSD. He described various attempts make this go faster, pointing out that some obvious solutions appeared

to improve some aspects of system peformance (i.e. **ls** appeared to be better) while profiling showed that overall system performance had not really improved.

"Make it work and then make it go faster" is an old Bell Labs axiom, perhaps we should add: "then show it really does go faster under all circumstances."

Item 12: 10.05am **A. Burns, University of Bradford**

A Comparison of the UNIX and APSE Approaches to Software

Abstract

An important aspect of the Ada project is the attempt to design and implement a standard support environment for the development and maintenance of Ada programs. A number of Ada Programming Support Environment (APSE) projects exist; many are using UNIX as a basis for the work and as the starting point for design. There are however many important differences between the use of software tools under UNIX and that envisaged for an APSE. This paper is concerned with the use of software tools and their interfacing. Comparisons between a UNIX and APSE approach are given.

This was another, much needed, introductory talk.

Item 13: 10.36am **Bill Weir, STC IDEC Ltd**

C Unit Test Harness

Abstract

Module testing is an important and often neglected area of software testing. Traditionally, it has tended to be a largely undocumented operation in which the programmer pokes data interactively at a module until he believes it is working satisfactorily. Studies have shown that tests conducted in this manner are seldom adequate in their coverage of program paths. It is hoped that, by automating much of the process, and by relieving the programmer of the drudgery of creating driver modules, collecting results, etc., more extensive testing at a module level will be encouraged, with consequent reductions in testing and debugging costs at a later stage of the software cycle.

A talk full of interesting ideas. Bill presented a system where the testing of routines can be formalised and automated. I felt that I need something like this, if only to aid regression testing. The main problem was the specification of tests.

Q. Are there any tools to help in building tests?

A. No, not at present.

Q. What is the availability of this?

A. Sorry, don't know.

☛ Coffee ☚

The session chairman was Teus Hagen.

Item 14: 11.26am **M. A. Rathwell, University of Bradford**

Distributed Decision Making under UNIX

Abstract

Distributed Decision Making attempts to meet the need for supporting tasks which involve cooperation and conflict between differentiated organisational units. A DDM system provides a mechanism for linking several decision support systems in an organisation, so enabling groups which are not necessarily linked in a hierarchical manner to cooperate with one another. This is particularly significant in planning operations which require information from different people dispersed throughout an organisation. It can enable independent decision makers to semi-automate their work, explanations for decisions can be made widely available, and the resolution of conflict between nodes with different interests and perspectives can be supported.

Item 15: 11.50am **A. R. Pell, University of Reading**

An Interactive Information Retrieval System for UNIX

Abstract

Many problems exist in office and information systems for which an appropriate solution is an information retrieval system. The aim of the first part of this project has been to build an easy-to-use interactive system. This has involved analysing the concepts and information structures needed, as well as considering the user interface. The emphasis throughout has been to construct the system in such a way that it is easy for a novice user to handle. Building on the established system, the second part of the project will involve experimenting with differing input devices such as touch screens, mouse input, trackballs, voice, etc.

Item 16: 12.15am **Theo de Ridder, IHBO 'de Maere**

Automatic Generation of Syntax Directed Screen Editors

Abstract

From a new effective and automatic error-recovery scheme for LALR(1)-parsers a program generator is developed that produces a syntax directed screen editor for any language specification written in LEX and YACC.

☛ Lunch ☚

Session chair was Jim McKie.

Item 17: 2.00pm **J. R. Nicol, University of Lancaster**

CRS — A Powerful Primitive For Resource Sharing in UNIX

Abstract

This abstract focuses on a resource sharing system which we call 'CRS' (Connect Remote Shell). CRS is layered on top of the UNIX operating system and provides a powerful set of network services. The environment in which CRS was developed formerly consisted of a number of PDP-11/44 minicomputers, each running UNIX. A user's computing activities were typically centered on one of these machines. Circumstances changed when we obtained a Cambridge Ring local area network, since we were then presented with the possibility of sharing the available computing resources.

Item 18: 2.37pm **Brian E. Redman, Bell Communications Research**

Honey Danber — The UUCP of the Future

Abstract

In 1978, Mike Lesk was considering a mechanism to aid in the administration of software on the growing number of computers running UNIX at Bell Labs. He envisioned a system that would automatically synchronise several machines with updates from a single source. At the time, no networking software existed upon which to build such a system, so he invented the UNIX to UNIX Copy program (*uucp*) to transfer files among machines. Little did he know that *uucp* would become the foremost file transfer and remote execution facility for untold years to come. That which was created as a temporary measure to get data from one UNIX system to another has endured through time as one of the most beleaguered, yet most critically required UNIX utilities.

Brian's talk centred on a recent re-write of the *uucp* suite of programs which will be available on System 5, release 2. The re-write was undetaken by an ad-hoc committee and the programs are a product of "software engineering" and not just hacked together. The result is a much more flexible, faster and better controlled *uucp*.

☛ Coffee ☚

Session chair was Keld Simonson.

Item 19: 3.45pm **P Tintel, Bell Telephone Manufacturing Company**

EURONIX: A UNIX-based System using European Natural Languages

Abstract

The UNIX system has gained enormous popularity. It is used in many places for software development, and it is beginning to be used in office environments. However, the average office worker is no computer specialist and he has other demands concerning the system then software developers.

The UNIX user interface as it is today has certain drawbacks for office applications and more specific for office applications in Europe. The manuals are not very readable for someone not familiar with UNIX; all communication with the user is performed in English (or American); and UNIX is not capable of working with the different European characters in a uniform way.

In the Euronix project, focus is on the second and third problems: EURONIX will communicate with the user in the user's natural language and will be able to handle in a uniform way the special European characters.

The implementation is to alter UNIX from working in ASCII to working in the TELETEX character set which can cope with all the European "funny letters". In addition, all messages from programs pass through a string data base in the user's natural language.

Item 20: 4.16pm **C. Roberts, Inf. Techn Task Force, EEC**

Standardisation of the National Character Sets

Abstract

A review of the European Standard policy for national character sets will be given. The way it fits in the current EEC programs will be discussed: the general options of setting standards in character sets, some examples of standards of character sets, as well as the implementation policy (some keyboard experiments).

Item 21: 4.45pm **Emrys Jones, Chairman EUUG**

EUUG Annual General Meeting

Abstract

This is the official general meeting to present the new constitution.

This meeting was held as a bootstrap device to get the new constitution off the ground. The preliminary constitution was passed and the new structure formally inaugurated.

☛ And so to Dinner at the Erica ☚

The idea of having a conference dinner was a good one. However, the red wine was awful. There were some after dinner speeches to make it taste better. Theo de Ridder made a speech which I have acquired in machine readable form. I am indebted to Jaap Akkerhuis for twisting his arm, leg or some other piece of anatomy for it.

Item 22: 9.45pm **Theo de Ridder**

After Dinner with Theo

Dear delegates.

In the past scientists used a very sensible language to express and exchange their ideas. The importance of that old dead Latin was that you had to be conscious of its fixed syntax and semantics in order to use it. At this moment I am permitted to make a speech in English, without much knowledge of its syntactic or phonetic structure. I dare to do so because in a more interactive situation no-one ever interrupts me with error messages whatever my mistakes.

Looking around in an UNIX environment the similarity between a programming language and a natural one is remarkable. There is not any formal syntactic or semantic description available and still programs are made and ported all over the world. In case of a programming error the system is kind enough not to complain about its exact type or place, it just gives you a wink that something went wrong. Isn't it a nice paradox that such an honourable human reaction is considered as typically user-unfriendly?

Let me go on with an anthropomorphic view on UNIX. I am aware that most of you do have a rather intimate relation with it. Using the statistical argument that for 90% you are male and will prefer the opposite sex, I conclude UNIX is female! Well gentlemen, what about your behaviour over the last decade? In any case you exploited her. Sometimes you even raped, suppressed or sold her. And some individuals had the courage to publish the insultant proposal to bring her in the public domain! In spite of certain parallel liberation movements in society UNIX was not able to free herself from historical bounds into an independent respectable creature.

There is a more philosophic and fundamental aspect of the human condition than being male or female, and that is being mortal. So, UNIX is not eternal. She must be in one of the binary states, alive or dead, or else she is instable in illness. It is hard to prove where she is. Her growth and continuing changes indicate liveliness. The exponential increase however reveals a disease like cancer. And finally the cult of these meetings, the existence of gurus, and the need for myths synthesize the declaration of a posthumous holiness.

After dinner it is fun to tell each other fairy tales. Maybe you need a tutorial example. I hope it will be simple to apply the following to your own business.

Once upon a time there was a princess called V6. She was considered small and beautiful by her people. Many a handsome academic prince came along to ask for her hand. But her mother, Ma Bell, locked her up to prevent disintegration of the empire. And so she became old and ugly in grim electronic towers. Her only pleasure was looking out of a window into the silicon valley and watching the play of the big cat AT&T with a lot of little licenced mice like ... you.

Day 3 — 18th April

Well, unlike yesterday, prolonged late night discussion kept me in bed to miss the first session. The session chairman was Joachim Wolff.

Item 23: 9.30am **Joe Carfagno, Central Services Organisation**

Using UNIX on a Large Software Project

Abstract

This talk is about the uses of the UNIX operating system in a large software project. Many different implementations and releases of the UNIX system, including the UNIX (1100) system, are used in a variety of ways from software development, testing, project management, site support, and others. This talk will show the versality, flexibility, and portability of the UNIX system.

Item 24: 10.05am **Ludo Vemmekens, Amdahl Nederland B.V.**

Large Systems UNIX: Opportunity for Innovation

Abstract

In bringing UNIX to the large mainframe world, there is a merging of two standards: the UNIX standard of openness, ease of use, ease of development, and the IBM large operating system standard of high reliability, security, performance, and support. The combined standards are a new challenge to the operating system products world.

This paper will discuss in detail the technical issues in making UNIX a viable mainframe operating system. These include reliability, production operations management, communications, memory management, full duplex support, database, security, support, compilers, applications,

coexistence with MVS, VM, and UNIX.

☛ Coffee ☚

I got to the hall just in time to see Andrew Hume, session chairman and contestant in the "hairy knees of the conference competition" make a small opening speech deploring the current fashion of being rude about UCB. More power to his knees.

Item 25: 11.18am **David Tilbrook, Imperial Software Technology**

The 5 Pitfalls of Interactive Graphics

Abstract

The NEWSWHOLE system was created almost 10 years ago. A video tape of it has been widely distributed and used to teach interactive graphics. One way it has been used is to examine the way in which it avoids the 5 pitfalls of interactive graphics: Boredom, Confusion, Discomfort, Frustration and Panic. This presentation will discuss those pitfalls and the design strategy used to avoid them.

The abstract does not point out the thing of which David is most proud. The system worked on a high resolution display and a different cursor shape was used to indicate different operations. The one that springs to mind is the Buddha which meant "please be patient, I am computing." This idea is a goody, it's one of those things which when you see you wonder why everyone doesn't use it, but of course, you have to have the idea first.

Item 26: 11.50am **Brian E. Redman, Central Services Organisation**

Behind Every Binary License is the UNIX Heritage

Abstract

Lately there seems to be some pessimism about the future of the UNIX system. Many who have watched its development from the earliest days feel that the system appears to grow corrupt and is no longer a model of innovation in operating system design.

Unix was originally designed by a talented fraternity with a clear and common vision for a better environment. Ever since the system has been redesigned by a diversity of people with different goals the tend to be less clear. UNIX has evolved from a simple, elegant model into one that is certainly complex and often seems convoluted. It no longer constitutes a statement of smallness, but appears to be growing unrestricted. ...

This was certainly the funniest talk of the conference, we hope to get a reprint. Contrary to popular opinion, Brian was never in the running for the hairiest knees of the conference award.

☛ Lunch ☚

The afternoon session was chaired by Mike Banahan

Item 27: 2.02pm **Andrew Hume, AT&T Bell Labs Computer Research**

Processes Considered as Files

Abstract

Tom Killian has implemented images of running processes as full and legitimate elements in the file system. The image for process nnnnn is accessed as '/proc/nnnnn'. *read*(2) and *write*(2) work normally except that some system data is write protected. Ioctl's include stopping and starting a process, masking out signals that cause a stop, and returning a file descriptor for the text file (for the symbol table).

This is a neat idea costing 4K bytes in the kernel.

Item 28: 2.08pm **Daniel Karrenberg, University of Dortmund**

University of Dortmund's Spooling system

The University of Dortmund have several machines running several versions of UNIX but have few printers. They also have no Local Area Network, such as an ethernet. The talk described the spooling system which has been set up to cope with these problems.

Item 29: 2.15pm **Andy Greener, Imperial Software Technology**

EUUG Benchmarks: Some Results

Abstract

The EUUG Bench mark tests have been run on a variety of VAXs running UNIX-5, BSD4.1, BSD4.2. Results are presented and discussed.

The tests were on VAXes.

Item 30: 2.29pm **Johan P. Moelaert, Twente University of Technology**

A Semaphore Implementation in UNIX

Abstract

For certain purposes the UNIX system lacks strength in its possibilities of interprocess synchronisation. Several processes waiting for one event require an abundancy of process control when implemented with signals, and this is not very reliable. Mutual exclusivity may be implemented using 'open' and — if this fails — 'create', but this is neither very elegant nor completely foolproof. The pipe mechanism offers a good instrument for synchronisation, but can only be used between processes that have hierarchical relationships. For these reasons we decided to implement Dijkstra's semaphores in the UNIX system.

This was an ingenious solution which added some system calls to do the semaphore user interface. The system uses the in-memory inode table to store state of the semaphore. The talk was the only one to actually put some C up on the screen, and for that reason alone, scored some points.

Item 31: 2.42pm **Neil Mayhew, Bleasdale Computer Systems**

Experiences With Implementing IBM Bisync in UNIX

Abstract

Bleasdale UNIX micros are used in a variety of situations commercial, scientific and academic, and there is now increasing demand for distributed processing in the form of connecting micros to existing mainframe and database facilities.

Bleasdale's first step towards meeting this demand was to promote file transfer facilities, including RJE (Remote Job Entry), using IBM 3780 Bisync.

Item 32: 2.55pm **Theo de Ridder, IHBO 'de Maere**

MABENCH: A Portable Benchmark Machine

Abstract

In the computer science education (HIO) department of our institution we developed a complete synthetic benchmark package BENCH. In BENCH it is possible to specify arbitrary workloads with any number of parallel processes. Scriptfiles can be made by editing or by running application oriented scriptfile generators. In the output all the characteristic performance values (throughput, response time, service time) are given in table format.

The MABENCH system is a small portable machine which can be connected to the system under test via normal terminal connections. The small processor (6809) then sends several scripts to the test machine while performing measurements.

Item 33: 3.10pm **Nick Nei, University of Glasgow**

Measuring the Disk I/O on a VAX

Abstract

This paper describes a project under way at Glasgow University to gather statistics about disk performance on a VAX running Berkeley UNIX 4.1. These results will be used to construct a stochastic model for the behaviour of the disk subsystem. We hope that by modifying the parameters on the model and studying the results we can discover new ways of improving the disk and file system performance.

It seems that the measurements show no really discernable statistical model which is applicable to disc performance.

☛ Coffee ☚

Item 34: 3.30pm **Panel Session chaired by David Tilbrook, IST**

Is There a Future for UNIX?

During this session, I had to go to catch a plane. Richard Hellier from the University of Kent kindly agreed to take notes and produce a report. So here it is. The panel was:

L. L. Crume, AT&T
R. Raglen-Kelly, Pyramid
K. McKusick, UCB
H. J. Thomassen, U of Nijmegen CS Dept.
M. Banahan, The Instruction Set Ltd.
A. Freed, IRCAM.

Each of the panel members made a short statement on the subject:

Does UNIX have a future, and, if so, which way is it going?

Larry Crume expects further developments in networking tools and support and much greater use of windowing software at every level; Many novice interfaces would be required in future. He also mentioned the unbundling of UNIX and the simplification of the licencing process.

Roger Raglen-Kelly was concerned about people hacking UNIX for the sake of UNIX itself. He thought that UNIX was already too big and that something should be done to arrest the growth. What he wanted to see: was better internal support for multiprocessing and true networking; functional and object-oriented programming languages & shells.

Kirk McKusick reminded everyone that the days of formal releases of code from Berkeley were over and that they were returning to being a research institute.

Adrian Freed stressed the importance of separating UNIX from UNIX-based applications; With the preponderance of commercial and industrial users of UNIX, he felt that only a minority of UNIX users cared what the underlying system was. All they are interested in is their Spreadsheet, DataBase or whatever.

H.J.Thomassen followed up this line, later amplified by Teus Hagen, and pointed out that the Academic Community was moving in a different direction from the business community.

Questions and comment were taken from the audience.

Emrys Jones introduced a point of information, namely that Computer Magazines & Journals, as a group, were now outselling "girlie" magazines. He asked,

Did the panel feel that the "hobbyist" section of the user community would ever have much impact on the UNIX community as a whole?

Due to the fragmentation of the enthusiast sector, no-one anticipated any significant developments from this area.

Teus Hagen pointed out that although business users may be moving one way, any individual user will still be free to follow his own path.

Another speaker emphasised that UNIX should be viewed as a way of doing things rather than as a static entity.

Following a prediction of researchers moving away from UNIX towards, say, expert and knowledge based systems, Eric Allman noted that all this new code would have to be developed somewhere, and probably on UNIX systems.

Kirk McKusick spoke of possible further developments of the UNIX kernel to support concurrent running for tightly-coupled multiprocessor systems.

Replying to a question on the conformity of AT&T products to /usr/group standards, Larry Crume said that AT&T were part of /usr/group and so their code would be 100% compatible with those recommendations.

Several speakers expressed concern that the UNIX tradition of distributing source code, even of the kernel, would not be upheld by the commercial users and software-houses. Larry Crume affirmed that AT&T would continue to supply source code with System V and its successors. Many attendees wanted some form of pressurisation on business users to distribute sources of their products. Before taking up the legal aspects of this topic, Mike Banahan observed that few business users care about source code; All they want is a tool for a job, he felt.

The next question was:

How can software developers protect their investment unless they restrict source?

Eric Allman, speaking for Britton-Lee, pointed out that small firms simply could not afford even one lawsuit to test a copyright case. He quoted "one week to liquidation" if his firm ever became involved in such an action. Only the giants, like AT&T, could afford such a case. Britton-Lee will sell the source, but at a high price.

There was then a brief discussion of the impact of APSE technology on the UNIX community. Several speakers feel that even if the project fails to achieve its goals, like Multics, it will contribute much to the industry.

In response to the alleged unchecked growth of the UNIX kernel, Kirk McKusick observed that size must be traded off against functionality, i.e. that a kernel with a given set of services must be of at least a certain size.

The final topic was the future of Inter-Process Communication, instigated by Dave Tilbrook. Larry Crume felt that, in the current absence of any formalism for describing IPC we still don't know which way to go. None of the panel members would be drawn on this one.

Eric Allman observed that IPC enables software developers to write code that runs in user space and then move it into the kernel if memory allows.

There was a general view that there must be a logical separation between particular applications and the IPC mechanisms they employ. IPC enables us to *communicate* rather than *convert* software. That is, when some new service becomes available we converse with that rather than its predecessor, and to keep the applications small.

This spawned a nostalgic side-discussion about the good-old-days of Version 6; Eric Allman observed that much of the "elegance" of that system arose from its implementation on a particular machine architecture, i.e. PDP-11, and that many of the techniques employed were simply the only ways to proceed on such a machine.

As a postscript, there were a few questions about the purpose, or lack of it, of having future EUUG meetings.

Endpiece

Well, that's it for another EUUG conference. The papers which constute the proceedings will be published separately.

Thanks are due many people who worked very hard to make the event a success. Hendrik-Jan Thomassen and George Rolf headed up a team of tireless workers. David Tilbrook did the programme organisation aided and abetted by Teus Hagen. The EUUG secretariat, Helen and Debbie, did their usual good job.

And the winner of the "hairy knees of the conference" award? Well, it was a close run thing But since there were only two competitors, and both were Australian, the winner must be Richard Grevis.

EUUG Nijmegen 1984 — A Report

Harold Cross

As UniForum was described as a zoo (more like a circus), EUUG brought to mind a peaceful gathering in the park (perhaps reading poetry aloud). There were approximately 300 attendees at the meeting. The technical content was high and the commercial influence was low-key. The conference was held at the University of Nijmegen. The auditorium was a lecture hall in design fitted with high-tech audio/visual apparatus. There was a vendor exhibition held in the promenade outside the auditorium and in a couple of modest sized rooms. Although the wares were uninteresting to me for the most part, there were machines to play with and give-a-ways of buttons, tee-shirts and plastic bags. The lack of a powerful exhibition in no way detracted from the meeting, it enhanced it. There were a couple of interesting terminals. The best was the M2150 from Microcolour Graphics. It has a 4096 color palette (16 at a time) with hardware (power of two) zoom, area fill, pan and a cross-hair cursor. The graphics resolution was 640x384. There is an independent text plane (80/132 x 80). The price was around $US3000.

The meeting began the evening before the first day of talks as people formed SIGs in the local hotel bars. The European UNIX community is not unlike its North American counterpart. It is populated by interesting and knowledgeable personalities (some more than others of course). Perhaps because of my perspective (but more likely because of the character of the meeting) the conference was excellent, technically rich and socially invigorating.

I listened to three quarters of the talks, attended a few more than that and missed a few entirely. I will describe some of the talks.

Emyrs Jones began the meeting by not giving the opening remarks in Dutch (apparently his general custom was to open in the native tongue).

Mike Kelly from Teletype talked about the 5620. (He also gave a short sales pitch on the 3B series of computers.) One of its features is that it doesn't require you to come up with another version of UNIX for your workstation (you just have to run a version that supports the 5620 on your host). It runs on standard system V from AT&T Technologies. It even runs on a VAX, "which is important today, but probably won't be tomorrow.", Mike says. A megabyte of memory makes a more usable system. He described layers, xt and demux functionally. Although he couldn't make a commitment, he wouldn't be surprised if there was a Berkeley port before long. They are looking at color, peripherals (disks), and compatibility with other UNIX systems. A questioner from the audience (from AT&T-BL) suggested a terminal ought to come with the capability to download it. Another asked why Mike was talking about peripherals. Next they'll put UNIX on it. Kelly replied that they certainly won't put UNIX on it, it's not a suitable architecture. Someone from England with a Canadian accent complained about non-interaction of the layers and the lack of ability to have multiple processes in a layer.

Kelly promised IPC within the 5620 by the next release and RS422 support within a year. Olivetti will be the exclusive distributors of the 3B line and terminals.

After the coffee break, Adrian Freed introduced a couple of talks on the future of UNIX at Amdahl and AT&T-BL. He advocated (I think) a merger of System V and Berkeley versions and voiced a plea for the availability of source.

The speaker from Amdahl talked about their product. There was a 13 MIP model and a 20 MIP attached processor model. Lacking was full duplex ASCII support. Their next product would be an implementation of System V. It would also have paging and vfork(). He mentioned that after the third port to new versions of UNIX, they used the Bell (sic) source and implemented their changes with ifdefs.

Next Larry Crume spoke of where AT&T-BL UNIX was headed in the future. They will continue to provide source, they will continue to provide portability. He said humbly that AT&T-BL has learned, although it took them a while to learn, from Berkeley. He had the right perspective in that Berkeley was a research organization and their output was to be looked at carefully. Some of their work would be incorporated, some of it served as models which would be reimplemented, other things were not necessarily relevant or useful. He mentioned that job control was changed slightly from 4BSD (cough) so user programs don't have to worry about it. With respect to other features, "We will move forward to implement those in an evolutionary fashion." He was concerned over the growth of commands' sizes. Future UNIX — A base, the kernel with a minimal set of drivers and utilities (libraries). There would be add-ons such as the C language and other utilities (grep, sort, awk ?). He listed the basic commands. Paging kept cropping up and was eventually addressed. It isn't available yet because AT&T-BL hasn't been satisfied with the performance of their implementation. He hopes that a satisfactory implementation will come out in third quarter of '84 (he hopes many things will be available then). It will provide a large address space and isolate the memory management architecture. There will be no application program changes and it will not hurt users who don't need paging. He also expected record and file locking a la the UNIX standards committee.

Lunch was served. I believe it was some sort of chicken along with a meatball soup and fries. Fortunately Dutch Heineken does not taste like swamp water.

Bill Murphy spoke very briefly offering to give people the answers they need. (Is that like giving people the answers they want?)

After the tea break Eric Allman continued with his analysis of databases. I didn't take any notes having spent the entire time trying to get the writing shelf for my seat/desk in place.

The next talk was about an interesting application of UNIX. It was to provide a database for kidney dialysis patients and to control the dialysis machines dynamically (e.g., to remove water at a variable rate over time.) Using the computer, new techniques were implemented which otherwise were impractical. The obvious fear from some of us in the audience was the inevitable "Out of blood, core dumped" message. But the machines were very sophisticated and provided many failsafes. Artificial kidneys were passed around. The speaker observed that touch screens were a good gimmick but not really useful for his application.

Andrew Hume talked about integration of various user services. For instance given a mail service and a calendar service, could one send a calendar as mail? Surprisingly often, this is not possible. He talked about his work with multiply typed objects and a bitmapped/mouse interface which overcame these deficiences.

At some point during the talks a notice appeared on the chalkboard. "This year's competition. NOT annoy Rob Pike with new switches for cat. BUT suggest an extension to C which will benefit the community??" The list of suggestions grew throughout the day and was anonymously erased the next morning.

We were on our own for dinner (thank goodness) and a few of us ate in a hotel restaurant. It was fantastic and not very expensive. I did not attend the bar SIGs that evening but dutifully retired to my room (10 minutes away in nearby Groosbeck) to finish some transparencies.

Tuesday morning began with a talk on dynamic profiling from Kirk McKusick. He covered the dos and don'ts of tuning as well.

Next was a discussion of the APSE and unix approaches to software tools. The problems of a large software project were discussed (>100 people, >50K lines, 30+ years lifetime). They were listed as:

1) divergence of intended program behavior
2) cost estimates exceeded
3) late delivery
4) logical errors/unreliable
5) high maintenance costs (> 70%)
6) duplication of effort

Next the outcome of the software crises was described. Better languages, better methodologies, and support environments. An APSE is a database comprising an information repository for the entire life cycle. Also, communication interfaces (user, system and tool interfaces) and a toolset — integrated set of tools for entire life cycle. The same kernel APSE (KAPSE) is implemented on various operating systems to make the environment look the same. Why a database?

1) tools need not know information representation
2) information can be added without affecting other tools
3) flexible attributes and relationships can be constructed
4) transactions
5) version control
6) integration

As always, there is the conflict between flexibility and structure.

The next talk was about a C Unit Test Harness. It performed automatic test driver generation using a test spec file and did execution logging. The spec must have predicted output. The test then is easily repeatable and can be used with a test coverage monitor (profiler).

After the morning break the following session included a talk about automatic generation of syntax directed screen editors. A bottle of fine Dutch beer was introduced. The beer represented students — full of energy. The bottle itself was the Dutch bureaucracy. The bottle was shaken and opened, the students' energy was impressive. The syntax directed editors are generated from the language definitions as represented in the scanner and parser. A generic editor module is linked to the lex and yacc outputs. The editor incorporates a cursor synched parser, as the cursor is positioned, the grammar is parsed. State information appears just below the cursor. There is a window for output. There is an interactive option which caused the editor to enter a dialogue. Some applications were discussed. A cobol editor ("Cobol is a nice language because it is mostly redundant"). A 'sh' editor. It was noted that the shell was not a language, the frustrated efforts to generate a BNF for it were discussed. Its inconsistency was described along with an interesting bug. (Try "<<'ls'" on your machine. We did on most of those in the vendor exhibit. The shell either died, with or without a core dump, or hung interminably. In one case a micro was unable to be rebooted after this experiment was performed.) The editor handled errors as best it could. If only one token was possible it was inserted. If more tokens could be used it suggested one. Otherwise it refused the token with an error message.

We were again subjected to lunch (something yellowish served over rice).

A simplistic network implementation was described. It is very much like the 'net' command which first appeared in UNIX 3.0 and suffers many of the same deficiencies. Such a limited approach can however be quite useful, although perhaps not completely satisfactory.

Brian Redman spoke at length about the next version of *uucp*. Europeans are concerned mostly about phone costs and therefore clamor for more efficient use of the phone line and direct X.25 connections.

Sometime after tea there was a general meeting of the EUUG. A draft constitution was voted in by show of hands and an initial executive committee was approved (I abstained). EUUG has unique problems due to geographic considerations. The idea of free student attendances based on personal recommendations was discussed (scholarships of a sort). The prospect of tutorials to subsidize the technical meetings was put forth. (By the way, EUUG does not pay the lecturer in a tutorial.) And in order to bring down the fees, unbundling of the registration was discussed. For instance, don't include lunch. It's interesting to note that EUUG is somewhat like USENIX was before Boston. They are about to struggle with the same problems of dealing with vendors and keeping the meeting down to a manageable size and keeping it technically rich and informal. I trust they can learn a few things from USENIX's tribulations.

Two talks followed emphasising the anguish Europeans have about character sets. The problem is that 128 is certainly not enough, 256 can be made to do. Unfortunately many programs believe characters are only 7 bits. The first talk was technical, the second was delivered by an EEC bureaucrat, perhaps to the wrong audience. Both talks had a message; if you think there is a rift between Bell and Berkeley, try getting the Europeans to agree on character set standards.

There was a EUUG sponsored dinner banquet at which several members masqueraded as scheduler states (runin, runrun, etc). There was a speech by Theo de Riddler in which he compared UNIX to a woman, prostituted, abused, etc. Then it was compared to a famous deceased religious figure whom many worship. As a gesture to the past (glorious in its modesty), Fifth Edition manuals were presented to Bill Murphy and Jim Kennedy of AT&T International.

At the bar afterwards there were many interesting discussions with various people. Several of us got into an extended discussion of the future of EUUG (and USENIX) relating to its size and purpose. There was discussion of uucp and ifdefing different algorithms for time and space tradeoffs. There was discussion of a source code stockroom administered by USENIX. Someone from AT&T-BL was very optimistic that all sorts of add-ons would be released soon. There was much interest in V8, the advice was to write letters. There were arguments about the proper placement of network routing algorithms. On the one hand it should be a separate command (mail 'route ist'!dt). On the other it was argued that it belongs in uucp (or mail). Along similar lines there was discussion of the feasibility of a dynamic routing database. How do you deal with an anarchic environment. Nothing will work for long. However any attempt is better than the current situation. There was no tequila and no Amaretto so we were stuck with beer and Grand Marnier. After the management kicked us out from the bar, the diehards reconvened in someone's room. The meeting lasted until after 5 am. I however dutifully retired so that I might prepare for a talk.

On Wednesday morning Joe Carfagno spoke about large systems. A very good presentation describing overall management of a two million line application. The talk covered development, customer interface, management tools, testing, etc. A lot of material covered smoothly and clearly. Interesting aspects of the project described are that it was successful, on time, reliable and efficient. Apparently a key factor was the rich UNIX environment.

After coffee Andrew Hume opened his session by briefly admonishing the current practice of criticizing Berkeley merely because it's fashionable to do so. There are plenty of valid criticisms to be made without resorting to aspersions as social rhetoric.

David Tilbrook talked about the five pitfalls of interactive graphics. Boredom, Confusion, Discomfort, Frustration and Panic. He showed a film about "Newshole" (an interactive news copy layout aid he had designed) to make his points.

The next talk was a replay of the "Behind every Binary License" rhetoric from UniForum '84.

After another astounding lunch there was a series of five minute talks on various topics. Andrew Hume talked on /proc in V8. Karrenberg from Dortmund discussed a spooling system implemented in a similar way and at the same time as Berkeley. Greener from Imperial Software Technology gave some EUUG benchmark results. System V is fairly close to 4.2BSD. Speeded up comets (15%) DO EXIST. Tummers from Twente (Holland) described a simple implementation of Dijkstra's semaphores.

Mayhew from Bleasdale gave a long talk on implementing IBM bisync on UNIX without the vpm. Ridder from Holland described an inexpensive and portable way to benchmark UNIX (or most other systems). It consisted of a 68000 master computer controlling a small network of 6809's feeding terminals. Nick Nei from Glasgow gave a fast and furious talk on modeling disk requests. Apart from an unexpected peak at 50Hz, there was no real insight into what is going on.

Then after tea there was a panel discussion of the future of UNIX. Each panel member gave a brief statement:

Larry Crume — Need UNIX to drive complex micros and loadable device drivers. Predicted people don't want to be given software, that it has to be packaged. Noted that the good old days are gone.

Robert Ragen-Kelly — Concerned that UNIX will be made to do things it wasn't intended to do. Urged that UNIX be kept small.

Kirk McKusick — Looking forward to receiving his degree. Said that Berkeley will move to function as a research facility. Predicted less formal releases.

Adrian Freed — Ought to be source distributions of everything.

Mike Banahan — People at one time were interested in UNIX itself. Now they're interested in what the system can do. Be careful of relying on a single piece of software.

H. J. Thomassen — UNIX will go the way of any commercial success. Away from universities towards businesses. Hopes that UNIX will provide good administrative packages. Growth of community is fragmented to points where a meeting is meaningless.

Discussion then was involved with who is still interested in UNIX in and of itself. The hobbyist market perhaps? The state of UNIX was compared to that of the automotive industry in the 40's. A good motor was available, what was needed was bodies and luxuries. Lots of discussion about pros and cons of the availability of source. Discussion of development environments. What about creating them for large teams by stepwise construction from working smaller environments? I've yet to see a really good panel discussion among intelligent level headed participants. This was no exception.

I conclude that the meeting was a grand success. It held my interest throughout and I came home with some ideas and many more contacts. It's too bad Europe is so far away, but I suspect the Europeans like it where it is.

Trivia Quiz

The following quiz was distributed at the Salt Lake City conference by Rob Pike. Prizes were awarded to the people with the most correct answers. The submission with the most correct answers (60) was from I. P. Stubbies (at team comprising David Tilbrook, Sam Leffler, and presumably others). Since they used a silly name and tried to put one over on the judges, they got a silly prize: a trophy labeled "world's best kibitzer." Jim McKie had the best score for an individual (57) and was awarded an authenticated 1972 DECtape containing UNIX Version 2. Finally, Ron Gomes had 56 correct answers and received an original engraved "Bill Joy" badge, which once belonged to Bill himself, from Sun Microsystems.

The answers to this quiz will appear in the next issue of *;login:*.

1. The source code motel: your source code checks in, but it never checks out. What is it?
2. Who wrote the first UNIX screen editor?
3. Using TSO is like kicking a [what?] down the beach?
4. What is the filename created by the original *dsw*(1)?
5. Which edition of UNIX first had pipes?
6. What is -=O=-?
7. Which Stephen R. Bourne wrote the shell?
8. Adam Buchsbaum's original login was sjb. Who is sjb?
9. What was the original processor in the Teletype DMD-5620?
10. What was the telephone extension of the author of *mpx*(2)?
11. Which machine resulted in the naming of the "NUXI problem"?
12. What customs threat is dangerous only when dropped from an airplane?
13. Who wrote the Bourne shell?
14. What operator in the Mashey shell was replaced by "here documents"?
15. What names appear on the title page of the 3.0 manual?
16. Sort the following into chronological order: a) PWB 1.2, b) V7, c) Whirlwind, e) System V, f) 4.2BSD, g) MERT.
17. The CRAY-2 will be so fast it [what?] in 6 seconds?
18. How many lights are on the front panel of the original 11/70?
19. What does FUBAR mean?
20. What does "joff" stand for?
21. What is "Blit" an acronym of?
22. Who was rabbit!bimmler?
23. Into how many pieces did Ken Thompson's deer disintegrate?
24. What name is most common at USENIX conferences?
25. What is the US patent number for the setuid bit?
26. What is the patent number that appears in UNIX documentation?
27. Who satisfied the patent office of the viability of the setuid bit patent?
28. How many UNIX systems existed when the Second Edition manual was printed?
29. Which Bell Labs location is HL?

30. Who mailed out the Sixth Edition tapes?
31. Which university stole UNIX by phone?
32. Who received the first rubber chicken award?
33. Name a feature of C not in Kernighan and Ritchie.
34. What company did cbosg!ccf work for?
35. What does Bnews do?
36. Who said "Sex, Drugs and UNIX"?
37. What law firm distributed Empire?
38. What computer was requested by Ken Thompson, but refused by management?
39. Who is the most obsessed private pilot in USENIX?
40. What operating system runs on the 3B-20D?
41. Who wrote *find*(1)?
42. In what year did Bell Labs organization charts become proprietary?
43. What is the UNIX epoch in Cleveland?
44. What language preceded C?
45. What language preceded B?
46. What letter is mispunched by *bcd*(6)?
47. What terminal does the Blit emulate?
48. What does "trb" stand for (it's Andy Tannenbaum's login)?
49. allegra!honey is no what?
50. What is the one-line description in vs.c?
51. What is the TU10 tape boot for the PDP-11/70 starting at location 100000 (in octal)?
52. What company owns the trademark on Writer's Workbench[tm] Software?
53. Who designed Belle?
54. Who coined the name "UNIX"?
55. What manual page mentioned Urdu?
56. What politician is mentioned in the UNIX documentation?
57. What program was *compat*(1) written to support?
58. Who is "mctesq"?
59. What was "ubl"?
60. Who bought the first commercial UNIX license?
61. Who bought the first UNIX license?
62. Who signed the Sixth Edition licenses?
63. What color is the front console on the PDP-11/45 (exactly)?
64. How many different meanings does UNIX assign to '.'?
65. Who said, "Smooth rotation butters no parsnips"?
66. What was the original name for *cd*(1)?
67. Which was the first edition of the manual to be typeset?
68. Which was the first edition of UNIX to have standard error/diagnostic output?

69. Who ran the first UNIX Support Group?
70. Whose Ph.D. thesis concerned UNIX paging?
71. Who (other than the obvious) designed the original UNIX file system?
72. Who wrote the PWB shell?
73. Who invented *uucp*?
74. Who thought of PWB?
75. What does *grep* stand for?
76. What hardware device does "dsw" refer to?
77. What was the old name of the "sys" directory?
78. What was the old name of the "dev" directory?
79. Who has written many random number generators, but never one that worked?
80. Where was the first UNIX system outside 127?
81. What was the first UNIX network?
82. What was the original syntax for "ls -l | pr -h"?
83. Why is there a comment in the shell source "/* Must not be a register variable */"?
84. What is it you're not expected to understand?

First Ever USENIX GO Tournament Results

Peter Langston

The First Ever USENIX GO Tournament was held at the Summer 1984 USENIX conference in Salt Lake City. The results and comments are given below.

Game Num.	Black Player	User Time	White Player	User Time	Approx. Score	Total Moves	Elapsed Time
1	NEMESIS	4:01	jim	23:06	~100	242	1:02:27
2	ogo	0:38	goanna	0:04	15.5	464	0:06:31
3	goanna	0:02	NEMESIS	7:23	~−175	325	0:38:45
4	jim	33:03	ogo	1:50	~−160	415	1:49:38
5	ogo	1:18	NEMESIS	9:36	−85.5	375	0:46:27
6	jim	12:18	goanna	0:01	>0	177	0:24:34
7	NEMESIS	7:37	goanna	0:02	~200	370	0:10:05
8	goanna	0:03	ogo	1:41	32.5	463	0:05:53
9	NEMESIS	7:25	ogo	1:04	85.5	375	0:11:13

Notes on the Games

The first six games formed a round-robin involving all the programs entered. These games were played between 2 p.m. and 6 p.m. (14:00 − 18:00) on Thursday, June 16th.

Game 1: NEMESIS vs. jim − NEMESIS wins
After 242 moves NEMESIS was beating jim quite handily. In a stereotypically Japanese gesture jim chose death over dishonor and made an illegal move (playing a suicide). Note that jim spent about six times as much time "thinking" as NEMESIS spent.

Game 2: ogo vs. goanna − ogo wins
This game was as fast as the first was slow. It became obvious from the first few moves that

ogo was living up to the symmetry of its name by playing mirror images of all of goanna's moves. Naturally this took little or no "thinking" time. Knowing that, it's a matter of no small wonder that goanna took even less time. When the dust settled ogo had a small lead.

Game 3: goanna vs. NEMESIS – goanna wins
This game had an unusual rhythm. NEMESIS would spend some time choosing a move and would print it (three seconds on the average) then goanna would immediately print a move (one one-hundredth of a second on the average). After 325 moves NEMESIS had a commanding lead but was apparently having trouble choosing a move. After a ten minute wait the judges peeked into the transcript file containing the communications back and forth between the programs and found the following dialogue (effectively):

REFEREE: your move?
NEMESIS: error 16: maxdepth=32, depth=32, movindex=56367, fatal=yes
REFEREE: your move?
NEMESIS: stack overflow; too many interesting possibilities – restarting
REFEREE: your move?
NEMESIS: would you like to play again?
REFEREE: your move?
NEMESIS: would you like to play again?
. . .

NEMESIS therefore lost by default.

Game 4: jim vs. ogo – ogo wins
Although ogo started this game with the symmetric play for which it had become famous, it was playing "normally" and was well in the lead (by about 160) at the point jim once again chose death over dishonor and made an illegal move (playing a suicide). Note that jim used about eighteen times as much user time as ogo did.

Game 5: ogo vs. NEMESIS – NEMESIS wins
NEMESIS lived up to its name by beating the heretofore unbeaten ogo in a game in which neither program dumped core or played illegally. The only really interesting feature was ogo's attempt to mirror NEMESIS's moves which eventually got it into trouble in the middle of the board. After 375 moves both programs passed and the score gave a solid lead to NEMESIS.

Game 6: jim vs. goanna – goanna wins
This game finally laid to rest the notion that jim was actually a very sophisticated personality simulation that only committed suicide to avoid losing by other means. After 177 moves jim made the usual illegal move and a count of the board showed that jim was winning at the time (although only marginally). Even in the face of such evidence, one diehard argued that jim's spirit must have been broken by the earlier losses, causing a misjudgement of the situation, "What could be more human?"

At this point there was a three-way tie for first place and no question as to who had secured last place. Each of ogo, goanna, and NEMESIS had won two and lost one; jim had lost three. The judges decided to play a round robin among the first place competitors and see what happened. Here's what happened:

Game 7: NEMESIS vs. goanna – NEMESIS wins
NEMESIS finally shows its true potential and wins by a whopping 200 points. Goanna, however, is not impressed and only spends 2 seconds total to ponder all 185 of its moves.

Game 8: goanna vs. ogo – goanna wins
Goanna took this game a little more seriously and spent 3 seconds of computer time to beat ogo by a narrow margin. Ogo's 101 seconds looked slow by comparison although both this game and game 2 were the only games to be limited by terminal baud rate rather than compute time. The only way to have sped this game up would have been to optimize curses...

Game 9: NEMESIS vs. ogo — NEMESIS wins

Once again ogo tries to mirror NEMESIS's moves and gets confused in the middle of the board. In many ways this game was identical to game 5; both games took 375 moves; in both games NEMESIS won by 85.5 points. Although this kind of repeatability is fairly common in experienced human GO players it's surprising to find it in these programs (especially ogo which was really written to test the referee program!)

RANKING

Place	Name	Author	Won	Lost	Comment
1	NEMESIS	Bruce Wilcox	4	1	Actually plays GO, ask for price
2	goanna	Bruce Ellis	3	2	Very fast
3	ogo	Peter Langston	2	3	Converted test program
4	jim	Hank Dietz	0	3	Slowly suicidal

Prizes!

Prizes were awarded by the tournament organizer and author of the referee program. The prizes were funded by the USENIX organization in return for no promotional considerations (but the tournament is named after USENIX, so...)

First Place

For some reason there were no First Place Computer GO Tournament trophies available in Salt Lake City (perhaps some other tournament had already bought them all). We did find a suitable trophy, however. It read "World's Best Golfer" and by crossing out the "lf" in the third word we were able to create the perfect trophy. Inside we found a little round sticker that said "The GO of UNIX".

Second Place

Similarly, the Second Place Computer GO Tournament trophies were all sold out, so a gorgeous black rubber bat, symbolizing speed, was awarded to goanna.

Third Place

Although there were plenty of Third Place Computer GO Tournament trophies available it was felt that the author of a program as sneaky as ogo really didn't deserve a trophy and that something else that embodied the spirit of nearly cheating would be more appropriate. A solid white Rubik's cube was found.

Fourth Place

It was not easy to find an award for this program's author especially in light of the controversy about jim's depressive behavior. It was felt that a much safer approach was to treat jim's misadventures as instances of bad luck, so a rabbit's foot and a wish for "better luck next year" were awarded.

Speaking of Next Year...

If there is interest in doing this again we will. If you'd like to make helpful suggestions, get further information, or generally encourage the proponents of serious fun send computer mail (*uucp*) to:

{decvax|ucbvax}!usenix!go

or send USnail Mail to:

Peter Langston
Lucasfilm Ltd.
P.O. Box 2009
San Rafael, CA 94912

Summer 1984 USENIX Conference Survey Results

As part of our efforts to get more feedback from members regarding our technical conferences, Randy Frank prepared a survey which was included in the Salt Lake City Conference registration packet. There were 1524 registrants at the conference, and 230 filled out and turned in the form. The results are tabulated below.

1. USENIX has traditionally held two technical conferences a year. Given your perception of the rate of technical development within the UNIX community, do you feel USENIX should continue with two conferences, or is one conference a year sufficient.

   50 one meeting a year; **159** two meetings a year; **21** don't care

2. Do you feel a vendor exhibition is desirable at the summer meeting which USENIX runs independently?

   173 exhibition; **18** no exhibition; **38** don't care

3. For the last several years USENIX has jointly sponsored the winter conference with /usr/group. In Dallas (January 1985) we plan on meeting at the date and location selected by /usr/group, but not on running a joint meeting. Assuming that USENIX meetings are held twice a year, do you feel USENIX should attempt to co-operate with /usr/group (meeting at the time and location selected by them), or should USENIX run an independent meeting at a different time and place.

   148 co-operate with /usr/group; **50** pick location and date independently; **22** don't care

4. At this conference USENIX allocated a separate budget item for professional audio/visual services. Do you think the money was well spent?

   185 money well spent; **7** money wasted; **28** no opinion

5. Did you notice an appreciable improvement in A/V over previous meetings?

   123 yes; **11** no; **76** no opinion

6. At this meeting the program committee elected to be selective about papers presented resulting in primarily a single track. Do you feel that two (or more) tracks are desirable in order to allow more papers to be presented even if the overall quality is lower?

   124 one track; **82** two or more tracks; **9** don't care

7. USENIX has, in recent meetings, scheduled a reception during one night of the meeting. Do you feel this is a good idea, or should this evening be left free for more BOFs, technical sessions, etc.

   183 reception; **19** no reception; **21** don't care

8. USENIX meetings have grown sufficiently large that the size is dictating locations where we can meet, number of sessions, etc. In order to keep meetings to a manageable size, would you favor imposing attendance limits even if it meant you might possibly be excluded from a future meeting?

   47 impose limit; **162** no limits; **13** don't care

The Board welcomes further comments, and will be reviewing these results at the next scheduled Directors meeting in the fall for guidance.

USENIX Conference Proceedings Available

Proceedings for the following USENIX conferences are available from the orginizations listed. California residents please add applicable sales tax. Payments must be in US dollars payable on a US bank.

Salt Lake City — Summer 1984, and Toronto — Summer 1983

Copies of the proceedings of the Salt Lake City Conference are available for $25 per copy, and of the Toronto Conference for $30 per copy. Add $15 per copy for overseas postage. Send your check or purchase order to:

USENIX Association
P.O. Box 7
El Cerrito, CA 94530

Washington DC UniForum Conference — Winter 1984

Copies of the proceedings of the UniForum Conference are available for $30 per copy, plus $20 per copy for overseas postage. They may be ordered from:

/usr/group
4655 Old Ironsides Drive, #200
Santa Clara, CA 95054

San Diego UNICOM Conference — Winter 1983

Copies of the proceedings of the San Diego UNICOM Conference are available for $25 per copy, plus $15 per copy for overseas postage. Send your check or money order to:

Software Tools Users Group
1259 El Camino Real, #242
Menlo Park, CA 94025

Summary of the USENIX Association Board of Directors Meeting

Hotel Utah, Salt Lake City
June 11th and 15th, 1984.

Present: Katz, Borden, Ferrin, Law, Nemeth, Scherrer, Wedel;
Guests: Bierley, DesHarnais, Frank, Johnson.

The Salt Lake City Conference

There were reports from Bierley and Frank concerning the organisation of the conference — everything seemed to be in good shape. Frank was installing an ethernet, and thought there would be 10 vendors attached and operational in a mixed protocol environment. The cable was donated by Belden, and hardware for taps was loaned by Interlan.

Proceedings for the conference were distributed to the Board — only two papers were missing, but Frank said the production required a very understanding printer, as copy went to press only two weeks prior to the meeting. The Board congratulated Randy Frank, Jay Lepreau, Spencer Thomas and Grant Weiler on a superb job — it was certainly setting new standards for future meetings.

There had been some problems with hotels — causing attendees to be shifted to the Little America Hotel. Both the Utah and the Sheraton had overbooked, as the deadline for holding rooms was one month before the start of the conference, at which time it appeared to the Hotels that the Association had considerably overestimated the attendance. Unfortunately a fair proportion of attendees make arrangements very near the time of the conference. Future contract with hotels will have to take this into account.

Other items were dealt with — estimates for Snowbird, the status of the conference budget, registration, the use of credit cards for the first time for registration, and tutorials.

Financial Report

Scherrer reported that the application for tax exempt status had, after a lot of red tape, been finally filed. It was decided to set aside in a separate bank account estimated taxes until the result of the application is known — this process may take on the order of a year or more. A financial statement from the start of the financial year through April was presented (5 months), and discussed — membership income has covered operating expenses for this period.

New Task Assignments

The results of the biannual elections for Officers and Directors of the Association were announced: President: Alan G. Nemeth, 292 for, 36 abstentions; Vice-President: Deborah K. Scherrer, 308 for, 21 abstentions; Secretary: Lewis A. Law, 301 for, 27 abstentions; Treasurer: Waldo M. Wedel, 296 for, 31 abstentions; Board Members: Bruce S. Borden, 184 votes, John Donnelly, 121 votes, Thomas Ferrin, 192 votes, Steve C. Johnson, 200 votes, Lou Katz, 241 votes, Brian E. Redman, 138 votes, Michael D. Tilson, 220 votes.

Following the elections Borden and Donnelly were leaving the Board; Wedel was to become treasurer, and Johnson and Tilson were joining the Board. This would require shuffling responsibilities — Wedel felt he could not continue to organise conferences and perform as treasurer, and a replacement was needed for Borden to organise tutorials. Decisions were deferred, but Wedel summarised the job of organising conferences — more of the work had been delegated to Judy DesHarnais (Conference Coordinator), an exhibition management company for the vendor exhibition, and to a professional audio/visual company for A/V during the conference, at least making the job tenable.

Ongoing Activities – Newsletter, Distribution Tape, Office, etc.

Brian Redman had agreed to be Technical Editor of the newsletter. Johnson thought that the future of *;login:* was bound up very much with the question of electronic publication and the possible publication of a Technical Journal. There was a lot of discussion, some centering around the question of responsibility for delivery of a publication distributed electronically. Katz and Johnson were named to a committee to look at all of these questions.

Betty Madden submitted a report on office operations. The Board felt that the office was too reticent in bringing problems to the attention of the Board – typical examples being the lack of a decent typewriter, and overtime being worked without recompense. These and other questions were referred to the office committee (Ferrin, Katz, Scherrer). Madden had gathered information on liability insurance – Law was to follow up. It was decided that a dbms should be acquired for the office computer – Katz was to follow up.

Scherrer said a call has gone out for submissions to the 84.1 distribution tape – several responses had been received. It was agreed that the cost of back issues of distribution tapes would be reduced to $75 – it was now much easier to reproduce them on the office 11/730.

The office computer, after a late arrival, is getting into use. Problems have to be dealt with by local Board members – there is no technical expertise in the office, and this has caused delays. More ports with modem controls are needed; Law will ship a Diablo printer owned by the Association to the office for use there; a software maintenance contract should be sought and executed.

Future Meetings

Before discussion of future meetings, it was thought appropriate to have a summary of the Washington meeting – Ferrin had received a financial statement and a check for monies owed. There had been discussions between Nemeth and Florio concerning possible arrangements for cross registration etc. at the Dallas meetings to be held in January 85; no formal agreement has yet been reached – Nemeth and Wedel were appointed as a committee to negotiate.

A considerable amount of time was spent discussing the Jan 85 Dallas meeting, and the various options with respect to schedule, program and format. It was agreed that a three day meeting would be held completely separate from the /usr/group UNIFORUM meeting, one day being for tutorials, and a further two for technical sessions. Tilson agreed to be responsible for tutorials, and some possible subjects and speakers were suggested.

A program chairman and program committee was needed for the Portland meeting (June 85) – it was hoped to find candidates at this meeting. Steve Bourne later agreed to be program chairman, with Stu Feldman, Dennis Ritchie and Greg Chesson as program committee.

Manuals

USENIX has finally received a license from Berkeley to print 4.2BSD manuals. This is going ahead – plates have been made; an order will be placed with the printer after a 45 day period to allow accumulation of orders.

Workshops

Workshops have been organised by: Nemeth on distributed systems at Newport, RI, Sept. 10th and 11th, 1984; Wedel on communications and networking at Golden, Co, Oct 11th and 12th; Katz on UNIX and graphics at Monterey, Ca, Dec. 13th and 14th. Attendance at each workshop will be limited to of the order of 100 people. Budgets were submitted and discussed.

Satellite Proposal

Lauren Weinstein proposed a solution to possible usenet problems by distribution of information by satellite, using the horizontal lines not seen during flyback of a television picture. He estimated a 64Kbit/s bandwidth would be available, and that it might be possible to get access via a public non-profit organisation. Financial support was required to demonstrate feasibility, and use of the Association's name to give credibility. After considerable discussion, both in the presence of Weinstein and later in the board meeting, it was felt that the Association should not take further steps that would lead to accepting responsibility for usenet — there were many pitfalls as yet little understood.

UUCP presentation

Karen Summers-Horten presented a report on ongoing work for obtaining and organising information concerning sites on the UUCP network. (This work is funded by the Association). Many "unknown" sites had been identified, and methods of obtaining the information were discussed. Horton asked to have access to the Association computer for storage of files — about half a megabyte of disc space is required — which was granted. Questions were asked about: security of information on the 11/730 (no problem); expense reports for the project; some technical issues and the schedule.

Network Communications

Various issues were discussed — the legal aspects of network communications in view of cases where systems had been shut down by police authorities and equipment confiscated. There was considerable further discussion of the Satellite proposal and the UUCP project — it was felt that the Satellite proposal was the first step towards becoming responsible for the usenet network, leading to ultimate responsibility, whereas the UUCP project was of short-term usefulness to members of the Association, without committment to support of the network.

It was agreed that legal advice was needed on these issues.

Review of the Salt Lake City meeting

There were 1524 registrants at the meeting, with approximately 700 attending tutorials. No surprises were expected in terms of expenses; the registration system appeared to have worked reasonably well, although generation of summary reports had problems.

The A/V services provided, (with the exception of Stu Feldman's talk — apologies to him) were better than at most previous conferences. The quality of slides in general was poor — other professional organisations send out specification material for slides — this should be done in the future.

Organisation of the vendor exhibition had gone smoothly. There was a lack of signs giving directions; attendance probably was not as good as usual, as it was not easy for conference attendees to wander into the exhibit area between papers as the technical part of the conference was in a different location from the exhibits; a vendor lounge was provided; it was felt that the level of technical expertise at the vendor booths was very high; the ethernet experiment had been a great success — 9 vendors were on line.

Feedback gained from talking to attendees and a quick perusal of the questionnaires handed out at the meeting indicated that the technical quality of the conference was thought to be high, availability of proceedings was appreciated, and that there had been no major snafus. Randy Frank and his associates had done a great job, which will be difficult to live up to in the future.

Other Business

A standing Nominating Committee was set up, with Borden, Donnelly and Frank being asked to serve. It was hoped that by doing this the committee would have more time to find people not only appropriate to serve on the Board, but also willing.

There was discussion of a topic that has recurred at several Board meetings — the need to find a person to deal with many of the day to day operating decisions of the Association, who would be of such caliber as to be able to handle some of the more time-consuming tasks presently having to be handled by Board members. It was also agreed that there should be considerable effort made towards getting non-Board members involved in the operations of the Association. Volunteers would be welcome.

Formal Actions of the Board

PROPOSED by Nemeth, seconded by Scherrer, that at Portland all vendors should be restricted to 4 booth spaces on initial sales, with additional unbooked spaces being made available after March 1st, 1985. For 7, absent 1 (Donnelly). Motion passed.

PROPOSED by Nemeth, seconded by Katz, that Law be authorised to expend up to $3K to purchase liability insurance for the Board. For 7, absent 1 (Donnelly).

PROPOSED by Scherrer, seconded by Borden, that the Association reduce the fee for back issues of the Distribution Tape to $75. For 6, abstaining 1 (Law), absent 1 (Donnelly).

PROPOSED by Katz, seconded by Scherrer, that the Association authorise an expenditure up to $3K for legal research on issues related to network usage. For 7, abstention 1 (Nemeth).

Local User Groups

The USENIX Association will support local user groups in the United States and Canada in the following ways:

- Assisting the formation of a local user group by doing an initial mailing for the group. This mailing may consist of a list supplied by the group, or may be derived from the USENIX membership list for the geographical area involved. At least one member of the organizing group must be a current member of the USENIX Association. Membership in the group must be open to the public.
- *;login:* will publish information on local user groups. Information on local groups giving the name, address (phone number and/or net address), time and location of meetings, special events, etc. is welcome.

Please contact the USENIX office if you need assistance in either of the above matters. Our current list of local groups follows.

The Front Range group meets about every two months at different UNIX sites for informal discussions.

Front Range Users Group
N.B.I., Inc.
P.O. Box 9001
Boulder, CO 80301

Attn. Steve Gaede
(303) 444-5710
hao!nbires!ceg

Dallas / Fort Worth UNIX User's Group
Advanced Computer Seminars
2915 L.B.J. Freeway, Suite 161
Dallas, TX 75234

Attn. Irv Wardlow
(214) 484-UNIX

There is an informal group that meets in the Washington, D.C., area every two months or so. The current contact for that group is:

Neil Groundwater
Analytic Disciplines, Inc.
Suite 300
8320 Old Courthouse Road
Vienna, VA 22180

(703) 893-6140
npg@lbl-csam

Unigroup is a non-profit organization in the New York City area for users and vendors of products and services for UNIX systems.

Unigroup of New York
G.P.O. Box 1931
New York, NY 10116

The UNIX Users of Minnesota meets on the first Wednesday of each month. The August meeting will be held at 7:00pm at Augsburg College. For information on times and locations contact:

UNIX Users of Minnesota
Carolyn Downey
(612) 934-1199

In the Atlanta area there is a group for people with interest in UNIX or UNIX-like systems:

Atlanta UNIX Users Group
P.O. Box 12241
Atlanta, GA 30355-2241

Marc Merlin (404) 255-2848
Mark Landry (404) 874 6037

USENIX Association
P.O. Box 7
El Cerrito, CA 94530

First Class Mail

FIRST CLASS MAIL
U.S. POSTAGE PAID
El Cerrito, CA 94530
Permit No. 87

Workshops

Trivia Quiz

System V Performance Enhancements

EUUG Meeting Reports